Untyped Confluence In Dependent Type Theories

Ali Assaf, Gilles Dowek, Jean-Pierre Jouannaud, Jiaxiang Liu

Untyped Confluence In Dependent Type Theories

Ali Assaf, Gilles Dowek, Jean-Pierre Jouannaud, Jiaxiang Liu

Google Inc, INRIA&ENS-Cachan, Université Paris-Saclay and École Polytchnique, Tsinghua University

INRIA Project Deducteam at //dedukti.gforge.inria.fr/

Abstract

We investigate techniques based on van Oostrom's decreasing diagrams that reduce confluence proofs to the checking of critical pairs in the absence of termination properties, which are useful in dependent type calculi to prove confluence on untyped terms. These techniques are applied to a complex example taken from practice: a faithful encoding in an extension of LF with rewrite rules on objects and types, of the calculus of constructions with a cumulative hierarchy of predicative universes above Prop. The rules may be first-order or higher-order, plain or modulo, non-linear on the right or on the left. Variables which occur non-linearly in lefthand sides of rules or in equations must take their values in *confined* types: in our example, the natural numbers. The first-order rules are assumed to be terminating and confluent modulo some theory: in our example, associativity, commutativity and identity. Critical pairs involving higher-order rules must satisfy van Oostrom's decreasing diagram condition with respect to their indexes taken as labels. Our use of decreasing diagrams yields a modular proof of confluence on open terms. Our encoding of the hierarchy of universes was obtained by using the MAUDE completion tool twisted to fit our needs. The obtained set of rules exploits all the sophistication of our confluence theorem.

1. Introduction

The two essential properties of a type theory, consistency and decidability of type checking, follow from three simpler ones: type preservation, strong normalization and confluence. In dependent type theories however, confluence and type preservation are needed to build strong normalization models; confluence is needed to show preservation of product types by rewriting, an essential ingredient of the type preservation proof; type preservation is needed to show that derivations issued from well-typed expressions are well-typed, an essential ingredient of the confluence proof. One can break this circularity in two ways : by proving all properties together within a single induction [13]; or by proving confluence on untyped terms first, and then successively type preservation, confluence on typed terms, and strong normalization. The latter way is developed here.

The confluence problem is indeed crucial for type theories allowing for user-defined computations such as in Dedukti [5] and now Agda [24]. Current techniques for showing confluence by using van Oostrom theorem for higher-order rewrite systems [29] are restricted to type theories in which the rules are left-linear, have development closed critical pairs, and do not build associativity and commutativity into pattern matching. But allowing for non-left-linear rules or for non-trivial critical pairs, and computing over non-free data structures whether first-order like sets or higher-order like abstract syntax, is out of scope of current techniques. Such computations are however present in Dedukti, whose main ambition is to serve as a common language for representing proof objects originating from different proof systems. Encoding these proof systems makes heavy use of the rewriting capabilities of $\lambda\Pi$Mod, the formal system on which Dedukti is based [4, 12].

In Section 6, we describe a rewrite-based encoding in $\lambda\Pi$Mod of the *Calculus of Constructions with Cumulative Universes* $\mathrm{CCU}^{\infty}_{\subsetneq}$, which uses Nipkow's higher-order rewriting, non-left-linear rules, and associativity and commutativity. $\mathrm{CCU}^{\infty}_{\subsetneq}$ is a generalization of the calculus of constructions with an infinite hierarchy of predicative universes above the impredicative universe Prop. Together with inductive types, it forms the core of the *Calculus of Inductive Constructions* as is implemented in the proof system Coq. Encoding inductive types in the style of Blanqui [7] is relatively simple [9], we only allude to it here. The major difficulty when encoding $\mathrm{CCU}^{\infty}_{\subsetneq}$ is the treatment of *universe cumulativity*, which needs to be rendered explicit. Existing encodings of universe cumulativity in $\lambda\Pi$Mod have limitations. In [2], the infinite universe hierarchy is encoded by a set of function symbols indexed *externally*, hence *infinite*. The rewrite based attempt in [3] is confluent on ground terms only, restricting its use to encode type systems, like Matita, which do not include universe polymorphism. Our encoding is confluent on terms with variables, hence can support Coq's universe polymorphism.

Our major contribution is developed in Section 5: a result reducing the Church-Rosser property of a $\lambda\Pi$Mod theory on untyped terms to typed critical pair computations, which goes far beyond the most advanced available technique based on development closed critical pairs [27], which cannot handle our example. This result is applied to the previous encoding in Section 6. It can be used more generally to show confluence of dependent type theories like those definable in Dedukti.

The main technical tool we use, recalled in Section 4, is van Oostrom's decreasing diagrams for labelled relations, which permit to prove confluence of rewrite systems that verify a kind of local confluence property called decreasing diagram [28]: local peaks need to be joinable by rewrites whose labels are smaller, in some well-founded sense, than those of the local peak. This technique provides a modular analysis of local peaks by reflecting the various components of a rewrite system in the labels. In the case of $\lambda\Pi$Mod, we classify the rules and equations in three categories: the functional ones inherited from the λ-calculus; a set of user-defined first-order rules and equations forming a Church-Rosser, terminating, normal rewrite system [17]; and a set of user-defined higher-order rules whose left-hand sides are patterns [22]. Our definition of higher-order rewriting on untyped terms adapts Nipkow's definition given for typed terms by replacing β-normalization with Miller's β^0-normalization [22], which, unlike β, is terminating on untyped terms. Some β-steps that are implicit in Nipkow's must therefore become explicit in our setting.

Obtaining Church-Rosser calculi by putting together different confluent systems is known to be difficult in presence of non-left-linear rules [1]. Further, confluence of arbitrary non-left-linear rules is never preserved in presence of a fixpoint combinator [19], which can itself be encoded in the pure lambda calculus. To eliminate this difficulty, variables having multiple occurrences in lefthand sides of user's rules are guaranteed to operate on homogenous algebraic terms by a syntactic assumption, *confinement*: by ensuring

that no redexes other than first-order ones may occur below a non-linear variable of a rule, confinement eliminates heterogeneous local peaks that would not have decreasing diagrams otherwise, like those occurring in Klop's fixpoint combinator example. Confinement appears therefore to be a crucial original concept, which is built directly in the type system of λΠMod.

Because of right non-linearities of the user's rules, and because β-rewriting may stack redexes that were previously disjoint, we analyze in Section 3 the local peaks of the relation union of first-order rewriting, β-rewriting at a set of parallel positions, and higher-order rewriting at a set of *orthogonal* positions. A related relation, *multi-step* rewriting, that extends Tait and Martin-Löf's definition of parallel reductions initially designed for the λ-calculus [6], is used by van Oostrom for, in particular, analyzing confluence in first-order orthogonal systems via permutation equivalence [27]. Both relations use labelling techniques for describing how the relation can be seen as a succession of elementary steps. Unlike van Oostrom's, our labelling technique is external to the terms involved in the rewriting step, hence makes orthogonal rewriting a ternary relation. It is original in the way it allows to naturally factor out an overlapping peak into a critical peak and further reductions taking place inside the associated substitution.

Using the rewrite category as the first label component of a rewrite is the basic technique used for analyzing *heterogeneous* local peaks, whose two steps use rules of different categories. The label's second component is used to solve *homogeneous* local peaks, whose two steps use rules of a same category implying that their labels' first components are identical. Those between first-order rewrites are closed because we assume their confluence and so are those between two parallel functional rewrites because they are known to be confluent. In the first case, the label's second component is the term to be rewritten compared in the associated termination order. In the second, it is a natural number provided by the completeness theorem for decreasing diagrams [18, 28]. Closing local *ancestor* peaks for higher-order orthogonal rewrites is built in the definition of orthogonal rewriting. Closing *overlapping* peaks involves higher-order critical pairs. Because of our definition of orthogonal reductions, overlaps may be *horizontal*, at parallel positions, but also *vertical*, at nested positions. Finally, the critical peak lemma enables as usual the lifting of decreasing diagrams from critical pairs to critical peaks of the orthogonal rewriting relation.

Despite its complexity, this paper is essentially self-contained.

2. λΠMod

Terms. Let Σ be a set of symbols equipped with a fixed *arity*, called the user's *signature*. As usual, symbols of arity 0 are called *constants*. This signature is partitioned into three sets, namely $\Sigma \stackrel{\text{def}}{=} \Sigma_{fo} \uplus \Sigma_{cd} \uplus \Sigma_{ho}$, whose elements are called *algebraic* symbols, *confined algebraic* symbols, and *higher-order algebraic* symbols. Let $\mathcal{X}, \mathcal{Y}$ be two additional disjoint sets sharing no symbol with Σ, whose elements are called *variables* and *confined variables*.

We consider the language defined by the grammar

$$\begin{array}{rcl} A & \stackrel{\text{def}}{=} & C, M \\ C & \stackrel{\text{def}}{=} & y \mid g(\overline{C}) \\ M, N & \stackrel{\text{def}}{=} & x \mid f(\overline{A}) \mid \lambda x : M.N \mid M\ N \mid \Pi x : M.N \mid \Pi y : C.A \\ & & \text{with } f \in \Sigma_{fo} \cup \Sigma_{ho}, g \in \Sigma_{cd}, x \in \mathcal{X}, y \in \mathcal{Y} \end{array}$$

Our grammar defines two *sorts*: expressions originating from the non-terminal C are called *confined terms* or *terms of sort* **confined**, those originating from the non-terminal M are called *non-confined terms* or *terms of sort* **non-confined**, and those in A, their union, just *terms*.

We use $|\_|$ for the size of a term, or a finite set, or of a list.

The *head* of a term is its outermost symbol. Terms of the form $\lambda x : M.N$ are *abstractions*. Terms of the form $\Pi x : M.N$ or $\Pi y : C.A$ are *products*. Terms of the form $(M\ N)$ are *applications*. Application associates to the right: we write $(u\ \overline{v})$ for $(\dots(u\ v_1)\dots v_n)$. Terms of the form $f(\overline{U})$, with $f \in \Sigma$, are *algebraic-headed*, they must satisfy $|\overline{U}| = arity(f)$. Terms built solely from the signature and variables are called *algebraic*. Algebraic terms built from the sub-signature $\Sigma_{fo} \cup \Sigma_{cd}$ are called *first-order*. Confined terms are particular first-order algebraic terms. As usual, we write f instead of $f()$ when $arity(f) = 0$.

Note that λΠMod's type constructors * and □ are constants from Σ, regardless of their specific role and despite the fact that Σ is called the user's signature.

We use: $\mathcal{V}ar(M)$ and $\mathcal{BV}ar(M)$ for the sets of variables of M occurring free and bound respectively ; z for fresh variables ; $=$ for the syntactic equality of terms, $\stackrel{\text{def}}{=}$ for definitional equality.

Typing rules The use of λΠMod's typing rules is marginal here. However, they are slightly different from those given in [12], because of confinement.

There are two forms of judgements, $\Gamma \vdash M : A$ meaning that the term M *has type* A in the context Γ, and $\Gamma \vdash$ meaning that the context Γ is *well-formed*. In a context, variables from $\mathcal{Y}$ must be given a confined type. Those from $\mathcal{X}$ must not. Here are a few important rules, see [12] for the (classical) others:

JP: Give all typing rules here.

$$\frac{\Gamma \vdash A : *;\ y \notin \Gamma;\ y, A \text{ confined (resp., } x \notin \Gamma;\ x, A \text{ non-confined)}}{\Gamma, y : A \vdash \text{ (resp., } \Gamma, x : A \vdash)}$$

$$\frac{f : \Pi x_1 : A_1. \cdots \Pi x_n : A_n.B \in \Sigma \qquad \Gamma \vdash \qquad \Gamma \vdash M_1 : A_1, \dots, \Gamma \vdash M_n : A_n\{x_1 \mapsto A_1, \dots, x_{n-1} \mapsto A_{n-1}\}}{\Gamma \vdash f(M_1, \dots, M_n) : B\{x_1 \mapsto A_1, \dots, x_n \mapsto A_n\}}$$

$$\frac{\Gamma, x : A \vdash M : B \qquad x, A, B \text{ not confined}}{\Gamma \vdash \lambda x : A.M : \Pi x : A.B}$$

$$\frac{\Gamma \vdash M : \Pi x : A.B \qquad \Gamma \vdash N : A \qquad A \text{ non-confined}}{\Gamma \vdash M\ N : B\{x \mapsto N\}}$$

$$\frac{\Gamma \vdash M : A \qquad \Gamma \vdash B : * \qquad A \equiv B}{\Gamma \vdash M : B}$$

As in all extensions of LF, the conversion $\equiv$ is generated by all rewrite rules, whether built-in or user-defined.

This type system does enforce a stratification over the sublanguages of objects, types and kinds, objects being typed by types, types by kinds, and kinds (among which $*$) by □, see [12]. User-defined symbols must come along with a type or a kind. Symbols in Σ_{fo} must either be type constants (of type $*$), or first-order function symbols, of type $\Pi x_1 : s_1. \cdots x_n : s_n.s$, where n is the arity of f and $s_1, \dots, s_n, s$ are type constants.

A major change wrt traditional type systems, including that of today's Dedukti, are the rules for typing abstractions and applications: terms of a confined type cannot be abstracted over or argument of an application, nor applied since confined types are constants from Σ_{cd}. Note that confinement assumptions are not needed when the rules are used bottom-up: they result then from the definition of (untyped) terms. They are needed, on the other hand, when the rules are used top-down. They are also needed if, as is actually customary, the rules serve defining untyped terms and typed terms at the same time, without any additional reference to a grammar of untyped terms.

The following assumption ensures then that the set of typable confined terms is the set of typable terms of a confined type:

Assumption 1: (i) $\forall f : \Pi\overline{x} : \overline{s}.s \in \Sigma_{fo} \cup \Sigma_{ho}\ \ (s \notin \Sigma_{cd})$
(ii) $\forall f : \Pi\overline{x} : \overline{s}.s \in \Sigma_{cd}\ \ (s, \overline{s} \in \Sigma_{cd})$

Positions. Positions in terms are words over the natural numbers, using $\cdot$ for concatenation, Λ for the empty word, $\geq_{\mathcal{P}}$ for the prefix order on positions (*below*), $\leq_{\mathcal{P}}$ for its inverse (*above*), $>_{\mathcal{P}}$ for its strict part, and $p\#q$ for $\neg(>_{\mathcal{P}} \vee \leq_{\mathcal{P}})$ (*parallel* or *disjoint*). We use $P \cdot Q$ for the set $\{p \cdot q \,:\, p \in P, q \in Q\}$. Identifying a position p with the singleton set $\{p\}$, we use: given a set of parallel positions P and an arbitrary set of positions Q, $Q >_{\mathcal{P}} P$ for $\forall q \in Q\, \exists p \in P\, (q >_{\mathcal{P}} p)$; Q_{min} for the largest subset of minimal positions of Q ; given a term M, $\mathcal{P}os(M)$, $\mathcal{FP}os(M)$, $\mathcal{VP}os(M)$ and $l\mathcal{VP}os(M)$ for the following respective sets of positions of M: all positions, the non-variable positions, and the positions of all variables, and linear variables respectively. A term M is linear if $\mathcal{VP}os(M) = l\mathcal{VP}os(M)$. We also denote by: $M|_p$, the subterm of M at position $p \in \mathcal{P}os(M)$; $M[\overline{N}]_P$, where $P \subset \mathcal{P}os(M)$ is a list of parallel positions st $|\overline{N}| = |P|$, the term obtained by replacing in M all subterms at positions in P by the coresponding term in the list of terms $\overline{N}$, using the simplified form $M[N]_P$ in case all terms in $\overline{N}$ are equal to N. We use $\mathcal{V}ar(M)$ ($\mathcal{BV}ar(M)$), for the sets of variables of M occurring free (bound).

Substitutions. Substitutions are *sort-preserving*, *capture-avoiding* homomorphisms, written $\sigma \stackrel{\text{def}}{=} \{z_1 \mapsto M_1, \ldots, z_n \mapsto M_n\}$, of domain $\mathcal{D}om(\sigma) \stackrel{\text{def}}{=} \{z_1, \ldots, z_n\}$ if $\forall i\, (z_i \neq M_i)$, such that M_i is a confined term iff z_i is a confined variable. The substitution σ is *algebraic-headed* if all M_i's are algebraic-headed. We use postfixed notation for the application of a substitution τ to a term s or a substitution σ, as in $s\tau$ and $\sigma\tau$, called *instances* by τ of s and σ.

Given terms s, t, u, v, computing the substitution γ, whenever it exist, such that $t = s\gamma$, in which case t is an *instance* of s, or $u\gamma = v\gamma$, in which case γ is a *plain unifier* of u, v, is called *plain pattern matching* and *plain unification* respectively. Unification is sorted: confined variables must be replaced by confined terms. We use $SU(u, v)$ for the set of plain unifiers of u, v. It is well known that a *most general unifier* σ exists for any two unifiable terms, that is $\forall \gamma \in SU(u, v)\, \exists \rho\, (\gamma = \sigma\rho)$. The word plain may be omitted.

Given a term u and substitution σ such that $\mathcal{D}om(\sigma) \cap \mathcal{BV}ar(u) = \emptyset$, we define the term $u[\sigma] \stackrel{\text{def}}{=} u[x_1\sigma]_{O_1} \ldots [x_n\sigma]_{O_n}$ where $O_i \stackrel{\text{def}}{=} \{o : u|_o = x_i \in \mathcal{D}om(\sigma)\}$. The associated operation on terms is *not* a homomorphism. Called *variable-replacement*, it is nothing but a convenient notation for multiple replacements at positions of free variables in a term. It is of course not compatible with renaming of bound variables in the term. Note that it coincides with instantiation when $\bigcup_{x \in \mathcal{D}om(\sigma)} \mathcal{V}ar(x\sigma) \cap \mathcal{BV}ar(u) = \emptyset$.

Computation rules. In type theory, computation is based on rewrite rules. We use $\longrightarrow$ for a rewriting relation on terms, $\longleftarrow$ for its inverse, $\longleftrightarrow$ for its symmetric closure, $\longrightarrow\!\!\!\!\!\rightarrow$ for its reflexive transitive closure, $\twoheadleftarrow\!\!\!\!\!\!\rightarrow$ for its symmetric, reflexive, transitive closure called *convertibility*, $\Longrightarrow$ for its parallel closure at a set of parallel positions, and $\otimes\!\!\Longrightarrow$ for its nested closure to be defined later. An element s such that $s \longrightarrow t$ for no t is in *normal form*. A *normal form* for t, written $t{\downarrow}$ is a term s in normal form such that $t \longrightarrow\!\!\!\!\!\rightarrow s$.

Definition 2.1. *A* rule *is a triple $i : l \to r$, whose possibly omitted index i is a natural number, and* lefthand side *l and* righthand side *r are terms of the same sort. A rule is* proper *if $\mathcal{V}ar(r) \subseteq \mathcal{V}ar(l)$ and* regular *if $\mathcal{V}ar(r) = \mathcal{V}ar(l)$. A variable is* left-linear *(resp.* right-linear *;* linear*) in $l \to r$ if it occurs exactly once in l (resp. r ; l and r). An* equation *is a pair of rules $i : l \to r$ and $j : r \to l$. We denote by $=_E$ the congruence generated by a set of equations E.*

In $\lambda\Pi$Mod, rule i comes along with an environment Γ in which the typable instances of l, r by a substitution γ have the same type. Our analysis of untyped reductions can of course omit them.

Different rules may share the same index. Rules are *algebraic* if both sides are algebraic. So are associativity and commutativity.

Definition 2.2. *A* plain rewriting system *is a set R of proper rules. A term s* (plain-) rewrites *to t at position p, written $s \longrightarrow^p_R t$ if $s|_p = l\sigma$ for some rule $l \to r \in R$, and $t = s[r\sigma]_p$.*

An equational rewriting system *is made of a plain rewriting system R and a set E of regular equations. A term s* rewrites *(*modulo E*) to t at position p, written $s \longrightarrow^p_{(R,E)} t$ (or $s \longrightarrow^p_{R_E} t$), if $s|_p =_E l\sigma$ for some rule $l \to r \in R$, and $t = s[r\sigma]_p$. An equational rewriting system (R, E) is* terminating *if the relation $=_E \underset{R}{\longrightarrow}$ has no infinite chain.*

A normal rewriting system *is made of a plain rewriting system R and a terminating equational rewriting system (S, E) whose rules are called* simplifiers*. Let $SE = S \cup S^{-1} \cup E$. A term s* (normal-) rewrites *to t at position p, written $s \longrightarrow^p_{R_{S_E\downarrow}} t$ if $s|_p = s|_p{\downarrow_{S_E}}$, $s|_p =_{SE} l\sigma$ for some rule $l \to r \in R$, and $t = s[r\sigma{\downarrow_{S_E}}]_p$.*

We write $s \longrightarrow^p_{(R,S,E)} t$ if $s \longrightarrow^p_{S_E \cup R_{S_E\downarrow}} t$.

The position or set of positions at which a rewrite takes place may be omitted, or replaced by a condition it satisfies, like $p\#q$ or $P \geq_{\mathcal{P}} q$. We may record various information in the upper index of the rewriting arrow. This applies to the normal form notation too.

A first essential remark is that all kinds of rewriting we just defined preserve our two sorts of confined terms and non-confined terms, under our definitions of a rewrite rule and a substitution. A second is that normal rewriting is not stable by substitution instance, since terms to be rewritten need to be normalized beforehand at the rewritten position. The situation is even worse with variable replacement which will always require dedicated lemmas.

Definition 2.3. *Given a* normal rewriting system (R, S, E)*, we call*
conversion: *a pair (s, t) such that $s \underset{R \cup S \cup E}{\twoheadleftarrow\!\!\!\!\!\!\rightarrow} t$,*
peak: *a triple (s, u, t) such that $s \underset{(R,S,E)}{\twoheadleftarrow} u \underset{(R,S,E)}{\twoheadrightarrow} t$,*
local peak: *a triple (s, u, t) such that $s \underset{i_{S_E\downarrow}}{\overset{p}{\leftarrow}} u \underset{j_{S_E\downarrow} \cup k_E}{\overset{q}{\rightarrow}} t$ where $i : l \to r, j : g \to d \in R, k : l \to r \in S$, and*
local cliff: *a triple (s, u, t) such that $s \underset{i}{\overset{p}{\leftarrow}} u \underset{j_{S_E\downarrow} \cup k_E}{\overset{q}{\rightarrow}} t$ where $i : l \to r \in E, j : g \to d \in R, k \in S$.*

A local peak or cliff is disjoint *if $p\#q$,* ancestor *if $q \geq_{\mathcal{P}} p \cdot \mathcal{VP}os(l)$ and* critical *if $q \in p \cdot \mathcal{FP}os(l)$ (the latter two comparisons may make no sense when $S \cup E \neq \emptyset$).*

A normal rewriting system (R, S, E) is Church-Rosser if every conversion (s, t) is normal-joinable*: $s \underset{(R,S,E)}{\twoheadrightarrow} =_E \underset{(R,S,E)}{\twoheadleftarrow} t$, and strongly normalizing (SN, or* terminating*) if $\longrightarrow_{(R,S,E)}$ terminates in E-equivalence classes of terms, i.e., $=_E \longrightarrow_{(R,S,E)}$ terminates.*

Plain rewriting and rewriting modulo are particular cases of normal rewriting when $S = E = \emptyset$ and $S = \emptyset$ respectively, allowing us to factor out many definitions. Cases where normal rewriting is necessary are rewriting modulo associativity, commutativity and identity [16], implemented in the programming system MAUDE [11], and higher-order rewriting[23]. The general framework is introduced in [17], improving over [21].

Plain rewriting enjoys two key properties implying that disjoint and ancestor rewriting peaks are always joinable [14]. More generally, given two normal rewriting systems (R, S, E) and (R', S', E'), we have the following easy joinability properties:

$$\underset{i_{S_E\downarrow}}{\overset{p}{\leftarrow}} \underset{j_{S'_{E'}\downarrow}}{\overset{q}{\rightarrow}} \subseteq \underset{j_{S'_{E'}\downarrow}}{\overset{q}{\rightarrow}} \underset{i_{S_E\downarrow}}{\overset{p}{\leftarrow}} \quad \text{if } q\#p, i \in R, j \in R' \qquad \text{(DP)}$$

$$\underset{i}{\overset{p}{\leftarrow}} \underset{j_{S'_{E'}\downarrow}}{\overset{q}{\rightarrow}} \subseteq \underset{j_{S'_{E'}\downarrow}}{\overset{p\cdot O}{\twoheadrightarrow}} \underset{i}{\overset{p}{\leftarrow}} \underset{j_{S'_{E'}\downarrow}}{\overset{p\cdot P}{\twoheadleftarrow}} \quad \text{if } \begin{array}{l} i : l \to r \in R, j \in R' \\ q \geq_{\mathcal{P}} p \cdot \mathcal{VP}os(l) \end{array} \qquad \text{(AP)}$$

where O, P are sets of parallel positions

Note that (DP) would not hold if the result of a normal rewriting step were normalized up to its root, as in [21].

(AP) specializes to (LAP) when $q \geq_{\mathcal{P}} p \cdot l\mathcal{VP}os(l)$, in which case $P = \emptyset$: the obtained property is a commutation diagram as is (DP). (DP) and (LAP) together generate an equivalence on reductions called *permutation equivalence*, which is analyzed thoroughly in [27].

(DP) generalizes to rewriting at sets of parallel positions P, Q such that $P \# Q$. (LAP) generalizes to rewriting at a set Q of parallel positions such that $Q \geq_{\mathcal{P}} p \cdot l\mathcal{VP}os(l)$. Both are proved from (DP) and (LAP) by induction on $|P| + |Q|$ and $|Q|$ respectively.

(AP/LAP) use plain rewriting at position p. Using instead rewriting modulo or normal rewriting would not give a valid property, since the modulo part below p could interact with the rewrite at q. There are cases, however, where there are no interactions, in particular when the modulo part cannot apply above q and does not need to apply below q, in which case we say that rewriting at p *preserves* the redex at q.

Definition 2.4 (Preservation)**.** *Given a normal rewrite system (R, S, E), a normal rewrite $u \xrightarrow{p} v$* preserves *a position $q \in \mathcal{P}os(u)$ such that $q >_{\mathcal{P}} p$ iff $u[z]_q \xrightarrow{p} w[z]_{p \cdot O}$ for some set of parallel positions O st $v = w[v|_{p \cdot O}]_{p \cdot O}$ and $\forall o \in O$ $(v|_{p \cdot o} =_E u|_q)$.*

Note that the occurrences of z in w need not be replaced by exactly the same term to get v. Note also that $u|_p$ and $v|_p$ are in S_E-norml form by definition of normal rewriting. This explains that $v|_{p \cdot o}$ and $v|_{p \cdot o'}$ may be different if $o \neq o'$, and E-equal to $u|_q$ rather than $(S \cup E)$-equal to $u|_q$.

The absence of interactions above q results in practice from signature considerations that forbid any application of an equation or simplifier on the path from p to q, while the absence of interactions below q results from the fact that the path from p to q goes through a linear variable of the lefthand side of rule used at p.

The linear ancestor peak property becomes:

$$v \underset{i_{S_E}\downarrow}{\overset{p}{\leftarrow}} u \underset{j_{S'_{E'}}\downarrow}{\overset{q}{\rightarrow}} w \subseteq \underset{j_{S'_{E'}}\downarrow}{\overset{p \cdot O}{\Rightarrow}} \underset{i_{S_E}\downarrow}{\overset{p}{\leftarrow}} \quad \text{(LAP)}$$

where O is a set of parallel positions,

if $\begin{cases} i : l \to r \in R, j \in R', \ q \geq_{\mathcal{P}} p \cdot l\mathcal{VP}os(l), \\ \text{rewriting at } p \text{ preserves position } q, \\ \text{and } =_E \subseteq =_{S' \cup E'} . \end{cases}$

Proof. By definition of preservation, $u[z]_q \xrightarrow{p} s[z]_{p \cdot O}$ and $\forall o \in O$ $(v|_{p \cdot o} =_E u|_q)$, hence $v|_{p \cdot o} =_{S' \cup E'} u|_q$, and thus $v|_{p \cdot o} \underset{j_{S'_{E'}}\downarrow}{\overset{\Lambda}{\rightarrow}} w|_q$, hence $v \xrightarrow{p \cdot O} s[w|_q]_{p \cdot O}$. Now $w = u[w|_q]_q$, hence $w \longrightarrow s[w|_q]_{p \cdot O}$ and we are done. □

(LAP) generalizes to rewrites at a set of parallel positions provided $u \longrightarrow^p_{i_{S_E}\downarrow} v$ preserves all positions in that set. In this context it is sometimes useful to write $\underset{j_{S'_{E'}}\downarrow}{\overset{p \cdot O}{\Longrightarrow}}$ instead of $\underset{j_{S'_{E'}}\downarrow}{\overset{p \cdot O}{\Rightarrow}}$.

We now turn to a last classical property of rewrites:

Definition 2.5. *Given a (rewrite) relation $\longrightarrow$ and an equational theory $=_E$, we say that* $\longrightarrow$ commutes over $=_E$ *if for all u, v, w s.t. $u =_E v \longrightarrow w$, there exists s such that $u \longrightarrow s =_E w$.*

2.1 Functional computations in $\lambda\Pi$Mod.

$\lambda\Pi$Mod comes equipped with two rewriting schemas, α-conversion and β-reduction:

$$\begin{array}{ll} \lambda x : U \,.\, M = \lambda z : U \,.\, M\{x \mapsto z\} & \text{if } z \text{ fresh or } z = x \quad (\alpha) \\ \Pi x : U \,.\, V = \Pi z : U \,.\, V\{x \mapsto z\} & \text{if } z \text{ fresh or } z = x \quad (\alpha) \\ (\lambda x : U \,.\, M)\, N \to M\{x \mapsto N\} & \quad (\beta) \end{array}$$

Substituting N to each occurrence of x in M involves renaming the bound variables of M away from those of N in order to avoid captures: (β) rewrites *modulo* α, and is accordingly denoted by $\longrightarrow_{\beta_\alpha}$ (in short, $\longrightarrow_\beta$). Note that α and β preserve sorts because, on the one hand a confined term cannot be abstracted over, and on the other hand, a term in argument position of an application cannot be confined. Hence, a reduction like $(\lambda z.z\ u) \underset{\beta}{\longrightarrow} u$ with u confined does not exist in the untyped syntax.

Major properties of β-rewrites are that: $\longrightarrow_{\beta_\alpha}$, hence $\Longrightarrow_{\beta_\alpha}$, is Church-Rosser modulo α ; commutes over $=_\alpha$, as well as over any equational theory generated by a set of algebraic equations whose non-linear variables are confined ; and enjoys (LAP).

But since disjoint redexes in a term may become stacked *residuals* after β-rewriting, (LAP) does not extend to parallel rewriting.

A particular case of repeated β-rewrites plays a key role here:

$$(\lambda \overline{z}.u\ \overline{x}) \ \to \ u\{\overline{z} \mapsto \overline{x}\} \quad (\beta^0, \text{as dubbed by Miller})$$

Beta^0-rewrites have a key property for our setting: by decreasing the size of terms, whether typed or not, they are confluent and terminating on all terms. Further, they need no α-conversion and enjoy the generalized form of (LAP).

As anticipated, we need to characterize the behaviour of β-reductions in presence of variable-replacement:

Lemma 2.6. *Let $u = (\lambda x.s\ t)$ be a term and σ a substitution st $x \notin \mathcal{D}om(\sigma)$. Then, $u[\sigma] \underset{\beta}{\longrightarrow} s[\sigma]\{x \mapsto t[\sigma]\}$.*

Proof. $u[\sigma] = (\lambda x.s\ t)[\sigma] = ((\lambda x.s)[\sigma]\ t[\sigma]) = (\lambda x.s[\sigma]\ t[\sigma])$ by assumption that $x \notin \mathcal{D}om(\sigma)$. Hence $u[\sigma] \underset{\beta}{\longrightarrow} s[\sigma]\{x \mapsto t[\sigma]\}$. □

2.2 First-order computations in $\lambda\Pi$Mod.

We assume given three sets of first-order algebraic rules: rules R_{fo}, *simplifiers* S_{cd} and equations E_{cd}. Simplifiers and equations have confined lefthand and righthand sides. Rules in R_{fo} are confined/non-confined depending on the sort of their both sides.

Assumption 2: (R_{fo}, S_{cd}, E_{cd}) is a terminating, Church-Rosser, normal rewriting system.

This assumption isolates the Church-Rosser property of (R_{fo}, S_{cd}, E_{cd}). It implies in particular that (S_{cd}, E_{cd}) is a terminating Church-Rosser system. To check it, we need [17]:

Assumption 3: E_{cd}-unification and $E_{cd} \cup S_{cd}$-unification are finite and complete.

A common example is provided by the two equations of associativity and commutativity for E_{cd} and the identity rule for S_{cd}.

In order to ensure preservation of other rewrites, we also need:

Assumption 4: Variables in rules from R_{fo} are either (i) confined or (ii) linear (on both sides).

This assumption is automatically satisfied by the rules from R_{fo} which are confined, since they have no non-confined variables.

By definition of normal rewriting, firing rules of R_{fo} requires pattern matching their lefthand side l modulo $S_{cd} \cup E_{cd}$. Since, however, the non-confined variables of l are linear, pattern matching can be assumed plain below them. It follows that first-order rewrites with rule $l \to r$ preserve non-confined redexes, if any, below l.

We end up with instantiation and variable replacement by *normal* substitutions, that is in normal form wrt $(S_{cd})_{E_{cd}}$.

Lemma 2.7. *Let* $u \xrightarrow[(R_{fo},S_{cd},E_{cd})]{q} v$ *and* σ *a normal substitution st* $\mathcal{D}om(\sigma) \subseteq \mathcal{X}$. *Then* $u\sigma \xrightarrow[(R_{fo},S_{cd},E_{cd})]{q} v\sigma$ *and* $u[\sigma] \xrightarrow[(R_{fo},S_{cd},E_{cd})]{q} v[\sigma]$.

Proof. Instantiation and variable replacement by non-confined terms do not impact $(S_{cd})_{E_{cd}}$-normal forms. □

Example 2.1 (extracted from Section 6)**.** *Consider the confined signature made of the constants* $0, 1$ *and the binary symbols* $+, \max, \uparrow$ *and* $\Uparrow$. *Letters* i, j *denote confined variables while* a *is non-confined.*

Let E_{cd} *be made of associativity and commutativity for* $+$, S_{cd} *be the identity for* $+$, *and* R_{fo} *be the following set of rules:*

$$\begin{array}{lllll} 1: & \max(i, i+j) & \underset{m_1}{\to} & i+j \\ 2: & \max(i+j, j) & \underset{m_2}{\to} & i+j \\ 3: & \mathrm{rule}(i, 0) & \underset{m_3}{\to} & 0 \\ 4: & \mathrm{rule}(i, j+1) & \underset{r_1}{\to} & \max(i, j+1) \\ 5: & \Uparrow(0, a) & \underset{l_1}{\to} & a \\ 6: & \Uparrow(i+1, a) & \underset{l_2}{\to} & \uparrow(i, \Uparrow(i, a)) \end{array}$$

Assumptions 3 and 4 are satisfied, since, in particular, the variable a *appears linearly in rules. For Assumption 2, we must show termination in AC equivalence classes and check that the ACZ-critical pairs are joinable by rewriting. Termination can be achieved easily by Rubio's fully syntactic AC path ordering [25]. This order works very much like Dershowitz's recursive path ordering, but has a specific status for AC operators that performs additional monotonicity checks. All symbols can have a multiset status, but* rule *whose status must be lexicographic while the precedence can be* $\Uparrow > \uparrow > \mathrm{rule} > \max > + > 1 > 0$. *Routine calculations then do the job. Checking the* ACZ*-critical pairs is routine too, there is only a trivial one between the two* max *rules.*

2.3 Higher-order computations in λΠ**Mod**

We assume now given a set of *higher-order* rules R_{ho}. Rules that contain functional symbols on either side must be in R_{ho}, as well as algebraic rules having a non-linear non-confined variable (such as $x \times 0 \to 0$ with a non-confined variable x on left lost on the right).

Higher-order computations result from two kinds of rules, depending on the structure of their lefthand sides, either higher-order patterns in Miller's sense [22], or algebraic expressions as in recursor rules [7] or non-regular algebraic rules as above. Miller's patterns require using some form of higher-order pattern matching to fire rules, while first-order pattern matching suffices for algebraic lefthand sides.

Definition 2.8 (Pattern [22])**.** *A* pattern *is a* Σ_{ho}*-headed term* L *in* β*-normal form whose all non-confined free variables occur in* pre-redexes*, that is, in maximal non-applied subterms of the form* $(X\ \overline{x})$, $n \geq 0$, *where* $\overline{x}$ *is a vector of* n *distinct variables bound above in* L. *A pre-redex is* trivial *if reduced to a variable.*

The lefthand side $\mathsf{rec}(S(x), X, Y)$ of one of the two classical recursor rules over natural numbers, is a pattern: algebraic terms are indeed patterns whose variables are not applied, *possibly* because they are first-order (X above may be, Y is not). In particular, all first-order algebraic terms are patterns in our sense.

Unlike in Miller's original definition [22], patterns must be Σ_{ho}-headed and need not be in η-long form: the lefthand sides of the above recursor rule would not be a pattern otherwise. More generally, the lefthand side of any recursor rule as defined in [7], is then a pattern. As already pointed out in [17], the notion of pattern is quite robust. We could indeed require that patterns are in normal form for (R_{fo}, S_{cd}, E_{cd}). This would be practically relevant, but is not needed here.

The main property of a pattern is that it generates by instantiation very specific β-redexes and reducts at predictable positions. Consider the subterm $(X\ \overline{x})$ at position q in a pattern L, and a substitution $\sigma = \{X \mapsto \lambda\overline{z}.u\}$. Then, $(L\sigma)|_q \longrightarrow\!\!\!\!\!\!\to_{\beta^0} u\{\overline{z} \mapsto \overline{x}\}$. So, the instance of a pattern by such a substitution β^0-reduces to a term obtained by replacing the subterms of the form $(X\ \overline{x})$ by terms of the form $u\{\overline{z} \mapsto \overline{x}\}$ where u is a term which does not contain any free occurrence of variables in $\overline{x}$.

Assumption 5: Lefthand sides of higher-order rules are patterns whose all non-linear variables are confined.

Note that linear variables can be either confined or non-confined.

The following assumption simplifies the coming confluence proof and diagrams and is satisfied by our example:

Assumption 6: Righthand sides of higher-order rules are not (i) variables, nor (ii) abstractions.

Nipkow's definition of higher-order rewriting based on higher-order pattern matching was elaborated for terminating computations [23]. It operates on terms in β-normal form and β-normalizes the result. Our definition for non-terminating computations operates instead on redexes in β^0-normal form and β^0-normalizes their contractum: higher-order rewriting is seen here as normal rewriting with $(R_{ho}, \beta^0 \cup S_{cd}, \alpha \cup E_{cd})$, inheriting its properties. In particular, as for normal rewriting, a term need not be in normal form wrt $(\beta^0 \cup S_{cd}, \alpha \cup E_{cd})$ in order to be rewritten at some position p. It needs to be in normal form up to p only. This subtle difference with Nipkow's definition is instrumental for enjoying (DP) and (LAP).

Let therefore $S = \beta^0 \cup S_{cd}$, $E = \alpha \cup E_{cd}$, and $SE = S \cup E$.

Lemma 2.9. *The equational rewriting system* (S, E) *is Church-Rosser and terminating.*

Proof. We interpret a higher-order term by the pair made of the multiset of its confined subterms first, and then its size. Confined terms are compared in the associated termination order, and sizes with the natural order on natural numbers. Termination follows from the facts that β^0 decreases the size of terms, and the order on confined terms is stable by instantiation, hence by α-conversion. Since critical peaks and cliffs are those of (S_{cd}, E_{cd}), the Church-Rosser property follows. □

This justifies our intuition of higher-order rewriting being an instance of normal rewriting. We say that u is *normal* if $u = u{\downarrow_{S_E}}$. We need naming positions of variables in a pattern:

Definition 2.10 (Fringe)**.** *The* fringe F_L *of a higher-order pattern* L *is made of three sets of parallel positions: its* plain fringe $F_L^{plain} \stackrel{\text{def}}{=} \{p \in \mathcal{FP}os(L) : L|_p$ *is a trivial pre-redex*$\}$*, its* functional fringe $F_L^{fun} \stackrel{\text{def}}{=} \{p \in \mathcal{FP}os(L) : L|_p$ *is a non-trivial pre-redex*$\}$*, and its* confined fringe $F_L^{cd} \stackrel{\text{def}}{=} \{p \in \mathcal{P}os(L) : L|_p$ *is a maximal confined term*$\}$.

Higher-order rewriting can now be formulated "à la Nipkow":

Definition 2.11. *Given a term* u *normal below* $p \in \mathcal{P}os(u)$ *and a higher-order rule* $i : L \to R \in R_{ho}$ *st* $\mathcal{V}ar(L) \cap \mathcal{BV}ar(u) = \emptyset$, *we say that* u *rewrites with* i *at* p, *written* $u \xrightarrow[i \in R_{ho}]{p} v$ *or simply* $u \xrightarrow[i]{p} v$, *iff* $u|_p \overset{\geq_{\mathcal{P}} F_L^{cd}}{\underset{E_{cd}}{\twoheadleftarrow}} = (L\gamma)\Downarrow \stackrel{\text{def}}{=} L\gamma{\downarrow}^{F_L^{cd}}_{(S_{cd})_{E_{cd}}}{\downarrow}^{F_L^{fun}}_{\beta^0}$ *and* $v = u[R\gamma{\downarrow_{S_E}}]_p$.

Note that all confined steps apply below the confined fringe: confined subterms of a non-confined term $X\gamma$ remain identical in $X\gamma$ and u. Let us then consider an algebraic pattern L. Since there are no applications in L, then $F_L^{fun} = \emptyset$ and $(L\gamma)\Downarrow = L\gamma\downarrow^{\geq_{\mathcal{P}} F_L^{cd}}_{(S_{cd})_{E_{cd}}}$. If L is the lefthand side of a recursor rule, then $u|_p = L\gamma$ as usual, hence higher-order rewriting coincides with plain rewriting in this case. This allows us to treat all rules in R_{ho} uniformly, including recursor rules if any.

Firing a higher-order rule involves two distinct algorithms – higher-order pattern matching and pattern matching modulo $S_{cd} \cup E_{cd}$– taking place below the functional fringe of the lefthand side pattern L for the first, and below its confined fringe for the second. Since no position can be below both fringes at the same time, this makes firing higher-order rules modular with respect to each matching algorithm.

This is true of unification too: $S_{cd} \cup E_{cd} \cup \beta^0 \cup \alpha$-unification of two patterns reduces to higher-order unification of patterns on the one hand, and $S_{cd} \cup E_{cd}$-unification of confined terms on the other hand. The reason is that confined terms can only be equated to other confined terms, otherwise unification must fail. This is yet another, very practical, implication of our notion of confined expressions.

Since β^0-normalization occurs below p only, normal rewriting is monotonic. Stability, on the other hand, requires normalization:

Lemma 2.12. *Let u be a term and σ a substitution, both normal, st $u \xrightarrow[i:R_{ho}]{p} v$. Then $u\sigma\downarrow^p_{S_E} \xrightarrow[i]{p} v\sigma\downarrow^p_{S_E}$ and $u[\sigma]\downarrow^p_{S_E} \xrightarrow[i]{p} v[\sigma]\downarrow^p_{S_E}$.*

Usual stability is recovered for a large class of substitutions:

Definition 2.13. *A substitution σ is* preserving *if $\forall x \in \mathcal{D}om(\sigma)$, x is not confined, $x\sigma$ is normal, and is not an abstraction.*

Algebraic-headed normal substitutions are therefore preserving.

Lemma 2.14. *Let u be a normal term and σ a preserving substitution. Then, $u\sigma$ and $u[\sigma]$ are normal.*

Proof. Since $x\sigma$ is not confined, the confined subterms of $u\sigma$ and $u[\sigma]$ are those of u or $x\sigma$, hence are normal. And since $x\sigma$ is not an abstraction, no β-redex, hence no β^0-redex, can be created. □

In the sequel, we manage, most of the time, to use preserving substitutions so that unwanted normalization steps do not pop up along computations. In particular, preserving substitutions allow us to recover important properties of rewriting modulo that are not true of normal rewriting in general:

Corollary 2.15 (Stability)**.** *Let $u \xrightarrow[R_{ho}]{p} v$ and σ a preserving substitution. Then $u\sigma \xrightarrow[R_{ho}]{p} v\sigma$ and $u[\sigma] \xrightarrow[R_{ho}]{p} v[\sigma]$.*

Lemma 2.16. *Let u normal such that $u \xrightarrow[i\in R_{ho}]{p} v$. Then v is normal.*

Proof. Let $w = u[z]_p$ for some fresh variable z, and $\sigma(z) = u|_p$, hence $u = w[\sigma]$. By assumption, w and $\sigma(z)$ are normal. Let now $\tau(z) \stackrel{\text{def}}{=} v|_p$. By definition of higher-order rewriting, $\tau(z)$ is normal. Since rules are sort preserving, $\tau(z)$ is non-confined. Further, $\tau(z)$ cannot be a variable nor an abstraction by Assumption 6(i,ii). Hence τ is preserving and $v = w[\tau]$ is normal by Lemma 2.14. □

Example 2.2. *Here is a rule (in untyped format) which is used in our encoding of* $\text{CCU}^{\infty}_{\subseteq}$ *described at Section 6:*

$$10: \ \pi(i, i+j+1, a, \lambda x.\uparrow(i+j, (b\,x))) \xrightarrow[p_4]{} \uparrow(i+j, \pi(i, i+j, a, b))$$

We see that rule 10 satisfies all our assumptions, in particular the non-confined variables a, b occur linearly on both sides of the rule. Note that the confined variables i, y occur non-linearly, on both the lefthand side and righthand side for i, and that the lefthand side contains a non-trivial pre-redex, thereby requiring higher-order pattern unification to fire the rule.

2.4 Commutation properties of higher-order rewrites

Higher-order rewriting enjoys preservation at all positions below the functional fringe of the lefthand side pattern of the rule used, provided they are headed by user-defined symbols:

Lemma 2.17 (Preservation)**.** *Let $u \xrightarrow[i:L\to R\in R_{ho}]{\Lambda} v$ and Q a set of parallel positions of $\mathcal{P}os(u)$ st $\forall q \in Q\,(q \geq_{\mathcal{P}} F_L^{fun}$ and $u(q) \in \Sigma_{fo} \cup \Sigma_{ho})$. Then $u[\overline{z}]_Q \xrightarrow[i]{\Lambda} w$ st $v = w[\{\overline{z} \mapsto v|_Q\}]$.*

Proof. By induction on $|Q|$. Let $Q = P \cup \{q = p\cdot o$ st $L|_p = (X\ \overline{x})\}$. By Definition 2.11, $u \overset{\geq_{\mathcal{P}} F_L^{cd}}{\underset{E_{cd}}{\twoheadleftarrow}} L\gamma\Downarrow$ for some γ. Hence $u|_q = (L\gamma\downarrow^p_{\beta^0})|_{p\cdot o} = (L\gamma|_p\downarrow_{\beta^0})|_o = ((X\gamma\ \overline{x})\downarrow_{\beta^0})|_o$. Since $u(q) \in \Sigma_{fo} \cup \Sigma_{ho}$, then $((X\gamma\ \overline{x})\downarrow_{\beta^0})|_o = X\gamma|_{o'}$ for some o'. It follows that $(u[z]_q)|_p = (X\theta\ \overline{x})\downarrow_{\beta^0}$, where $\theta(X) \stackrel{\text{def}}{=} X\gamma[z]_{o'}$. Since X is linear in u, then $u[z]_q = L\theta\downarrow_{\beta^0}$. Therefore, $u[z]_q \xrightarrow[i]{} w$ with $v = w[\{z \mapsto v|_q\}]$. We conclude by induction hypothesis. □

There are two kinds of commutations, whether some term $u[\sigma]$ rewrites concurrently to $v[\sigma]$ and $u[\tau]$, or successively to $v[\sigma]$ and $v[\tau]$. The first case corresponds to (LAP), while the second is a *permutation*. Note that all occurrences of $x\sigma$ are rewritten at once. These properties are of course related to preservation. In the case of preservation, there is a single rewrite from a term u to a term v at some position p, and a position q below p which is expected to be the position of some redex that will then occur in v as well. In both commutation and permutation properties, preservation is built-in by writing the term to be rewritten as the variable-capturing instance of some other term u rewriting to v.

Lemma 2.18. *Let u be a normal term and σ a preserving substitution such that $u \xrightarrow[i\in R_{ho}]{p} v$ and $x\sigma \xrightarrow[j\in R_{ho}]{\Lambda} x\tau$ for all $x \in X \subseteq \mathcal{D}om(\sigma)$. Then, $u[\sigma] \xRightarrow[j]{O} u[\tau] \xrightarrow[i]{p} v[\tau] \xLeftarrow[j]{Q} v[\sigma] \xleftarrow[i]{p} u[\sigma]$ for some sets O, Q.*

Proof. By Lemma 2.16, v is normal. Let $x \in X$. Since σ is preserving, $u[\sigma]$ and $v[\sigma]$ are normal by Lemma 2.14. Since σ is normal, τ is normal by definition of higher-order rewriting. Further, $x\tau$ is not confined since being a reduct of $x\sigma$ and is not an abstraction either by Assumption 6(i,ii). Therefore $u[\tau]$ and $v[\tau]$ are normal by Lemma 2.14 again. It follows that $u[\sigma] \xrightarrow[i]{p} v[\sigma]$ and $u[\tau] \xrightarrow[i]{p} v[\tau]$ by Lemma 2.12. The other rewrites are justified by letting $O = \{o : u|_o \in X\}$ and $Q = \{q : v|_q \in X\}$. □

Corollary 2.19. *Let u a term and σ a preserving substitution such that $u[\sigma]$ is normal, $u \xrightarrow[i\in R_{ho}]{p} v$, and $\sigma \xRightarrow[j\in R_{ho}]{O} \tau$. Then, $\exists P, Q$ such that (LAP) $u[\tau] \xrightarrow[i]{p} v[\tau] \xLeftarrow[j]{Q} v[\sigma]$ and (Perm) $u[\sigma] \xRightarrow[j]{P} u[\tau] \xrightarrow[i]{p} v[\tau]$.*

Proof. We prove (LAP), the proof of (Perm) being similar. Let $\theta(x) \stackrel{\text{def}}{=} x\sigma[\overline{y}]_{O_x}$ for all $x \in X$. We define $u' \stackrel{\text{def}}{=} u[\theta]$, $v' \stackrel{\text{def}}{=} v[\theta]$, $\sigma'(\overline{y}) \stackrel{\text{def}}{=} x\sigma|_{O_x}$ and $\tau'(\overline{y}) \stackrel{\text{def}}{=} x\tau|_{O_x}$. Then $u[\sigma] = u'[\sigma']$ and $v[\tau] = v'[\tau']$. Since $u[\sigma]$ is normal, so are θ, u' and σ'. Since v is normal, and θ is preserving, v' is normal by Lemma 2.14. Hence $u' \xrightarrow[i]{p} v'$. Further, $y\sigma' \xrightarrow[j]{\Lambda} y\tau'$ for $y \in \mathcal{D}om(\sigma')$. We conclude by Lemma 2.18. □

3. Orthogonal higher-order rewriting

A parallel rewrite step is a sequence of reductions at a set P of parallel positions, (DP) ensuring that the result does not depend upon a particular sequentialization of P. An orthogonal rewrite step is a sequence of reductions at some set P of positions, both parallel and nested, (DP) and (LAP) ensuring that the result does not depend upon a particular sequentialization of P. Our definition is based on splitting a term s into a term u and a preserving substitution σ such that $s = u[\sigma]$, therefore ensuring that (DP), (LAP) and (Perm) apply. Then u is rewritten at some set of orthogonal positions O, while σ is rewritten at a family of orthogonal positions $Q = \{Q_x\}_{x\in\mathcal{D}om(\sigma)}$, indexed by the domain of the substitution σ. The term u and the capturing substitution σ are not unique, provided they define the same term $u[\sigma]$ and sets of positions in it. The fact that the resulting term is then the same is a major property of orthogonal rewriting, and expresses the fact above that it does not depend upon a particular sequentialization of its set of positions. A particular case is of course obtained when O is a set of parallel positions and Q is the emptyset, or vice versa, resulting then in parallel rewriting. An important attribute of our formulation is that no hypothesis is made on the underlying rewrite relation, which can be plain, modulo or normal, and, more precisely, higher-order as is the case here. Another distinctive feature is that our relation is ternary: the orthogonal rewriting arrow is decorated with a label that internalizes the inductive history of its construction.

The idea of orthogonal rewriting appears in the literature under at least two different names, *parallel reductions* and *multi-step rewriting*. Parallel reductions, defined by induction on the term structure, were introduced by Tait and Martin-Löf to show confluence of the pure λ-calculus [6]. Multi-step rewriting generalizes the construction for both concrete and abstract rewriting relations. All these generalizations are extensively studied in [27], where they are used for analyzing orthogonal rewrite relations as well as, more generally, orthogonal rewrite steps of non-orthogonal rewrite relations, whether operating on first-order terms, higher-order terms or term-graphs. The idea of labelling rewrites is not new either, although the major trend in the literature is to incorporate the history of reductions in the rewritten term, rather than making the relation ternary, by decorating the rewrite system itself instead of the rewrite relation. We refer to [27] for a precise account of the literature and extensive bibliographic notes.

A major trait of both orthogonal and multi-step rewriting is that the underlying rewrite relation need not be orthogonal. Rewrite rules may have critical pairs, as well as repeated variables in their lefthand sides. Only the sequence of steps used in an orthogonal or multi-step rewrite needs to be orthogonal. The fact that it is still possible to analyze the Church-Rosser property of the underlying relation by studying its orthogonal extension relies on two properties: confinement, which takes care of repeated variables in lefthand sides of the rules, and the internalization of its construction, via a label in the case of orthogonal rewriting, that appears to be instrumental, as we shall see, to factor out critical peaks from two diverging orthogonal rewrite steps.

Definition 3.1. Orthogonal rewriting *with higher-order rule i at positions in O, written $\overset{O}{\underset{i}{\otimes\!\!\Longrightarrow}}$, is the smallest relation containing $\overset{O}{\underset{i}{\Longrightarrow}}$, and closed under* vertical product *(in short, product), written $u[\sigma] \overset{P\otimes_u Q}{\underset{i}{\otimes\!\!\Longrightarrow}} v[\tau]$, defined as*

(i) $u \overset{P}{\underset{i}{\otimes\!\!\Longrightarrow}} v$;

(ii) σ is a preserving substitution st $Q = \{Q_x\}_{x\in\mathcal{D}om(\sigma)}$ and $\forall x \in X\ (x\sigma \overset{Q_x}{\underset{i}{\otimes\!\!\Longrightarrow}} x\tau)$, abbreviated as $\sigma \overset{Q}{\underset{i}{\otimes\!\!\Longrightarrow}} \tau$;

(iii) $P \otimes_u Q \overset{\text{def}}{=} P \cup \{o\cdot q : x\in\mathcal{V}ar(u), u|_o = x \text{ and } q\in Q_x\}$.

Note that $P \otimes_u Q$ is defined for empty sets P, Q (by Q being empty, we mean that Q is a set of empty sets), $P = \emptyset$ corresponding to monotonicity and $Q = \emptyset$ to stability by preserving susbtitutions. Orthogonal rewriting reduces to the identity if P, Q are both empty, and to parallel rewriting if P is a set of parallel positions and Q is empty or vice versa.

Note further that $(P \otimes_u Q)_{min}$ is in general different from P_{min}, since there may be no position from P on a path from the root of u to some variable $x\in X$. This implies that our product construction is both horizontal and vertical, and that using a single rewrite step $u \overset{p}{\underset{i}{\longrightarrow}} v$ would define the very same binary relation between terms.

Note also that $u[\sigma]$ and $v[\tau]$ are normal below all positions in $(P \otimes_u Q)_{min}$ under assumption (ii) by Lemmas 2.14 and 2.16 – whithout that assumption, the definition of $P \otimes_u Q$ would be more complicated, since normalization steps would need to be reflected in (iii). We occasionally omit u in $\otimes_u$.

Using non-capturing substitutions in this definition would raise difficulties, because writing $u\sigma$, instead of $u[\sigma]$ implies that no variable of $x\sigma$ for $x \in \mathcal{V}ar(u)$ can be bound above in u, hence constraining u. There is a way out, though, by splitting $u[\sigma]$ into a term t and a substitution τ such that $t\tau \underset{\beta^0}{\longrightarrow\!\!\!\!\!\rightarrow} u[\sigma]$, and replacing in the definition $u[\sigma]$ by $(t\tau)\downarrow_{\beta^0}$. For example, if $\mathcal{D}om(\sigma) = \{x\}$ and $u|_p = x$, we can take $t \overset{\text{def}}{=} u[(x\ \overline{y})]_p$ and $\tau(x) \overset{\text{def}}{=} \lambda\overline{y}.x\sigma$. The obtained substitution τ is an abstraction, hence does not satisfy the assumptions of Lemmas 2.14 and 2.16. Although we believe that the entire theory can be carried out without these assumptions, this provisional choice makes the technical developments much easier.

Example 3.1. *Consider the rewrite system $\{f(f(x)) \to x\}$. Then $f(f(f(g(f(f(x)))))) \overset{\{\Lambda,1111\}}{\otimes\!\!\Longrightarrow} f(g(x))$.*

Product extends to substitutions by writing $\sigma[\gamma] \overset{P\otimes_\sigma Q}{\underset{i}{\otimes\!\!\Longrightarrow}} \tau[\theta]$. Then P is a set of sets of positions, while Q can be considered as a set of sets of positions indexed successively by the variables of $\mathcal{D}om(\sigma)$, then those of $\mathcal{D}om(\gamma)$, or, equivalently, by a set of positions indexed by their union. A routine check shows:

Lemma 3.2. *Let u, σ, O, P, Q st $O \subseteq \mathcal{P}os(u)$ and $\forall x \in \mathcal{D}om(\sigma)(P_x \subseteq \mathcal{P}os(x\sigma))$. Then, $O \otimes_u (P \otimes_\sigma Q) = (O \otimes_u P) \otimes_{u\sigma} Q$.*

Proof. Follows directly from the associativity of concatenation for positions. □

We now show a key property of orthogonal rewrites: the result of an orthogonal rewrite depends only on the input term u and the set P of orthogonal positions used, but not on their decomposition into a given product. To this end, we show that the term to be rewritten can always be decomposed into a term and a substitution whose frontier is defined by the set $(P \setminus P_{min})_{min}$.

Lemma 3.3. *Let $P \subseteq \mathcal{P}os(u[\sigma])$ for some linear term u and $O = \mathcal{FP}os(u) \cap P$. Then, $P = O \otimes_u Q$ for some Q.*

Proof. Take $Q_x \overset{\text{def}}{=} \{q : o\cdot q \in P \wedge u|_o = x\}$. □

This lemma will be mostly used as a way to define Q when given P and u, in which case we will write $P \overset{\text{def}}{=} O \otimes_u Q$, using indeed the equation as an implicit definition of Q.

Lemma 3.4. *Let $u \overset{P}{\underset{i}{\otimes\!\!\Longrightarrow}} v$, $u' \overset{\text{def}}{=} u[\overline{z}]_{(P\setminus P_{min})_{min}}$, $u'[\gamma] \overset{\text{def}}{=} u$ and $P = P_{min} \otimes_{u'} Q$. Then, $u'[\gamma] \overset{P_{min}\otimes_{u'} Q}{\underset{i}{\otimes\!\!\Longrightarrow}} v'[\theta] = v$.*

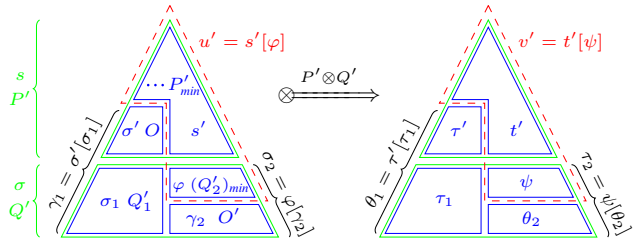

Figure 1. Proof of Lemma 3.4 with $P' \neq \emptyset$

Proof. By induction on the sum of length of positions in the set P. There are three cases according to P.

If $u \xrightarrow[i]{P} v$, then $Q = \emptyset$, the result holds with $u' = u$ and $v' = v$.

Otherwise, $u = s[\sigma] \overset{P' \otimes_s Q'}{\underset{i}{\Longrightarrow}} t[\tau] = v$. If $P' = \emptyset$, then $t = s$ and $\sigma \overset{Q'}{\underset{i}{\Longrightarrow}} \tau$. By induction hypothesis, $\sigma = \sigma'[\gamma] \overset{Q'_{min} \otimes_{\sigma'} Q}{\underset{i}{\Longrightarrow}} \tau'[\theta] = \tau$. By Definition 3.1, $u = s[\sigma] = s[\sigma'[\gamma]] \overset{\emptyset \otimes_s (Q'_{min} \otimes Q)}{\underset{i}{\Longrightarrow}} s[\tau'[\theta]] = s[\tau] = v$. By Lemma 3.2, $\emptyset \otimes_s (Q'_{min} \otimes_{\sigma'} Q) = (\emptyset \otimes_s Q'_{min}) \otimes_{s\sigma'} Q = P_{min} \otimes Q$. Hence $u'[\gamma] = (s[\sigma'])[\gamma] = s[\sigma'[\gamma]] \overset{P_{min} \otimes Q}{\underset{i}{\Longrightarrow}} s[\tau'[\theta]] = (s[\tau'])[\theta] = v'[\theta]$, concluding this case.

Otherwise, $P' \neq \emptyset$, this case being depicted at Figure 1. Let $s' = s[\overline{z}]_{(P' \setminus P'_{min})_{min}}$ where $\overline{z}$ is a vector of fresh variables, hence $s = s'[\sigma']$ for some σ'. Let now $P' = P'_{min} \otimes_{s'} O$ and $Q' = Q'_1 \uplus Q'_2$, where $Q'_1 >_{\mathcal{P}} P'_{min}$ and $Q'_2 \# P'_{min}$. Accordingly, we split σ as its restrictions σ_1 and σ_2 to respectively the variables of s which occur below P'_{min}, and those which do not. Let τ_1 and τ_2 such that $\sigma_1 \overset{Q'_1}{\underset{i}{\Longrightarrow}} \tau_1$ and $\sigma_2 \overset{Q'_2}{\underset{i}{\Longrightarrow}} \tau_2$. Hence $\tau = \tau_1 \cup \tau_2$. Let now φ such that $u' \overset{\text{def}}{=} s'[\varphi] = (s'[\sigma])[\overline{z}]_{(Q'_2 \setminus (Q'_2)_{min})_{min}}$, where $\overline{z}$ is a (new) vector of fresh variables. Note that $u' = (s[\sigma])[\overline{z}]_{(P \setminus P_{min})_{min}}$. Let finally $\sigma_2 = \varphi[\gamma_2]$ for some γ_2 and $Q'_2 = (Q'_2)_{min} \otimes_{\varphi} O'$.

By induction hypothesis, $s = s'[\sigma'] \overset{P'_{min} \otimes O}{\underset{i}{\Longrightarrow}} t'[\tau'] = t$ and $\sigma_2 = \varphi[\gamma_2] \overset{(Q'_2)_{min} \otimes O'}{\underset{i}{\Longrightarrow}} \psi[\theta_2] = \tau_2$. Remark that $u = s[\sigma] = s'[\sigma'[\sigma_1] \cup \varphi[\gamma_2]] = u'[\gamma]$, defining $\gamma \overset{\text{def}}{=} \gamma_1 \cup \gamma_2$, with $\gamma_1 \overset{\text{def}}{=} \sigma'[\sigma_1]$. Likewise, $v = t[\tau] = t'[\tau'[\tau_1] \cup \psi[\theta_2]] = v'[\theta]$, where $v' \overset{\text{def}}{=} t'[\psi]$ and $\theta \overset{\text{def}}{=} \theta_1 \cup \theta_2$ with $\theta_1 \overset{\text{def}}{=} \tau'[\tau_1]$.

Finally $u' = s'[\varphi] \overset{P'_{min} \otimes (Q'_2)_{min}}{\underset{i}{\Longrightarrow}} t'[\psi] = v'$. Further, $\sigma'[\sigma_1] \overset{O \otimes Q'_1}{\underset{i}{\Longrightarrow}} \tau'[\tau_1]$, hence $\gamma \overset{(O \otimes Q'_1) \cup O'}{\underset{i}{\Longrightarrow}} \theta$. Since $P_{min} = P'_{min} \otimes (Q'_2)_{min}$, we get the result by choosing $Q = (O \otimes Q'_1) \cup O'$. □

Because splitting P into P_{min} and Q depends solely on P as a set, the result of an orthogonal rewrite step is entirely determined by its source and set of orthogonal positions:

Corollary 3.5. *Let* $s[\sigma] = t[\tau]$, $s[\sigma] \overset{P}{\underset{i}{\Longrightarrow}} v$ *and* $t[\tau] \overset{P}{\underset{i}{\Longrightarrow}} w$. *Then* $v = w$.

These results mean that permutation equivalence (we refer to [27] for a precise definition) is built in the definition of orthogonal rewriting. Then, orthogonal rewriting inherits many properties of higher-order rewriting:

Corollary 3.6. *Let* $s =_E u \overset{P}{\underset{i}{\Longrightarrow}} v$. *Then* $s \overset{P}{\underset{i}{\Longrightarrow}} t =_E v$.

Corollary 3.7. *Let* u *a term and* σ *a preserving substitution st* $u[\sigma]$ *is normal,* $u \xrightarrow[i \in \beta \cup R_{fo} \cup S_{cd} \cup R_{ho}]{P} v$ *and* $\sigma \overset{O}{\underset{j \in R_{ho}}{\Longrightarrow}} \tau$. *Then,* $\exists P, Q$ *st (LAP)* $v[\sigma] \overset{P}{\underset{j}{\Longrightarrow}} v[\tau] \xleftarrow[i]{p} u[\tau]$, *and (Perm)* $u[\sigma] \overset{Q}{\underset{j}{\Longrightarrow}} u[\tau] \xrightarrow[i]{p} v[\tau]$.

Proof. We prove (LAP), (Perm) being similar. The case where $i \in R_{ho}$ follows by induction on $|O|$ based on Lemma 3.4. The case where $i \in R_{fo} \cup S_{cd}$ is similar to the proof of Corollary 2.19 (LAP), because higher-order rewriting preserving sorts, rewriting σ to τ cannot generate a confined simplification in $u[\tau]$ and $v[\tau]$. If $i = \beta$, let $u|_p = (\lambda x.s\ t)$. Then, $v[\sigma] = u[\sigma][s[\sigma]\{x \mapsto t[\sigma]\}]_p$ by Lemma 2.6, and since $x \notin \mathcal{D}om(\sigma)$, then $v[\sigma] \overset{P}{\underset{j}{\Longrightarrow}} w = u[\tau]$ $[s[\tau]\{x \mapsto t[\tau]\}]_p$ with $P = (\Lambda \otimes_u O) \otimes_{u[\sigma]} ((\Lambda \otimes_s O) \otimes_{s[\sigma]} (\Lambda \otimes_t O))$. Finally, $u[\tau] \xrightarrow[\beta]{p} w$ by Lemma 2.6. □

We now consider a kind of converse. Given an orthogonal rewrite step $s \overset{P}{\underset{i}{\Longrightarrow}} t$, can we arbitrarily split s into u and σ and P into O and P' such that $u[\sigma] \overset{O \otimes_u P'}{\underset{i}{\Longrightarrow}} t$? This is of course not possible in general, we need to take care of splitting away from all redex patterns involved in the orthogonal step (the notion of redex pattern is not defined here, but is intuitively the part of a redex that is replaced –and not copied– when applying a rewrite rule):

Lemma 3.8. *Let* $u \overset{P}{\underset{i:L \to R}{\Longrightarrow}} v$, *and* Q *a set of parallel positions st* $\forall q \in Q\ (u(q) \in \Sigma_{fo} \cup \Sigma_{ho})$ *and* $\forall p \in P\ (q \not\leq p \cdot \{o : \Lambda \neq o <_{\mathcal{P}} F_L\}))$. *Let* $s \overset{\text{def}}{=} u[\overline{z}]_Q$, $s[\sigma] \overset{\text{def}}{=} u$, $O \overset{\text{def}}{=} P \cap \mathcal{FP}os(s)$ *and* $P \overset{\text{def}}{=} O \otimes_s P'$. *Then,* $s[\sigma] \overset{O \otimes_s P'}{\underset{i}{\Longrightarrow}} v$.

Proof. The case $P = \emptyset$ follows from the definition. Otherwise, we proceed by induction on $|u|$. There are three cases:

1. $Q = \{\Lambda\}$. Then $s = z \in \mathcal{X}$ and $\sigma(z) = u$. Then $Q_z = P$.
2. $\Lambda \in P$, hence $P_{min} = \{\Lambda\}$, and wlog $\Lambda \notin Q$.
   Let $u_2 \overset{\text{def}}{=} u[\overline{z}]_{(P \setminus \{\Lambda\})_{min}}$ and $u \overset{\text{def}}{=} u_2[\theta_2]$. By Lemma 3.3, $P = \{\Lambda\} \otimes_{u_2} P_2$. By Lemma 3.4, $u_2[\theta_2] \overset{\{\Lambda\} \otimes_{u_2} P_2}{\underset{i}{\Longrightarrow}} v_2[\gamma_2] = v$.
   Let now $(P \setminus \{\Lambda\} \cup Q)_{min} = O_1 \uplus Q_1$ where $Q_1 \subseteq Q$ maximal and $O_1 \subseteq ((P \setminus \{\Lambda\}) \cap O)_{min}$. Let also $u_1 \overset{\text{def}}{=} u_2[\overline{z}]_{Q_1}$ and $u_1[\theta] \overset{\text{def}}{=} u_2$. Since, by assumption, $\forall q \in Q_1\ (u(q) \in \Sigma_{fo} \cup \Sigma_{ho})$ and $q \not\leq_{\mathcal{P}} F_L$, then $u_1 \xrightarrow[i]{\Lambda} v_1$ for some v_1 st $v_2 = v_1[\theta]$ by Lemma 2.17.
   Let $\theta_1 = \theta[\theta_2]$, $\gamma_1 = \theta[\gamma_2]$ and $P_1 = \emptyset \otimes_\theta P_2$. By definition of orthogonal reductions, $\theta_1 \overset{P_1}{\underset{i}{\Longrightarrow}} \gamma_1$ and $u_1[\theta_1] \overset{\{\Lambda\} \otimes_{u_1} P_1}{\underset{i}{\Longrightarrow}} v_1[\gamma_1]$. By easy calculations, $u = u_1[\theta_1]$, $v = v_1[\gamma_1]$ and $P = \{\Lambda\} \otimes_{u_1} P_1$. If $O_1 = \emptyset$, we are done. Otherwise, we recursively cut at the remaining positions of Q.
   Let $z \overset{\text{def}}{=} u_1|_o$ for some $o \in O_1$. Then $z\theta_1 \overset{(P_1)_z}{\underset{i}{\Longrightarrow}} z\gamma_1$. Let also $Q_z = \{q : o \cdot q \in Q\}$, $\sigma_1(z) \overset{\text{def}}{=} z\theta_1[\overline{z}]_{Q_z}[\sigma_2]$ and $(P_2)_z = \{p \in (P_1)_z : p \not\geq_{\mathcal{P}} Q_z\}$. By Lemma 3.3, $(P_1)_z = (P_2)_z \otimes_{z\sigma_1} (P_3)_z$. By induction hypothesis, $z\theta_1 = z\sigma_1[\sigma_2] \overset{(P_2)_z \otimes_{z\theta_1} (P_3)_z}{\underset{i}{\Longrightarrow}} z\tau_1[\tau_2]$.
   Let $z' \overset{\text{def}}{=} u_1|_o$ for some $o \in Q_1$, $\sigma_1(z') \overset{\text{def}}{=} z'$ and $\sigma_2(z') \overset{\text{def}}{=} z'\theta_1$. Lemma 3.2, closure of orthogonal rewrites under product, and Lemma 3.4 yield the result.

3. If $\Lambda \notin P \cup Q$, then, by induction hypothesis, the result holds for the immediate subterms u_i of u wrt to the positions P_i and Q_i belonging to $\mathcal{P}os(u_i)$. Putting back the head symbol of u gives then the result by monotonicity of orthogonal rewrites. □

3.1 Nested higher-order critical peaks

Generating the minimal *nested critical peaks* that reduce the confluence of higher-order orthogonal rewriting to the existence of decreasing diagrams for these peaks requires computing the overlaps of two orthogonal rewriting steps. The lefthand sides of the rules they use may overlap horizontally, as in parallel critical pairs, but also vertically, forming vertical chains of overlaps. Unlike traditionnal critical pairs, vertical overlaps between two lefthand sides may take place inside the substitution associated with a partial vertical chain, while self-overlaps involving the same lefthand side must be disallowed. Computing these overlaps requires then some bookeeping, both in terms of substitutions and overlapping positions. This is the role of the notion of *seed* introduced now:

Definition 3.9 (Seeds)**.** *Given two rules $k : L \to R, l : G \to D$ from R_{ho}, the set $p\mathcal{S}_l^k$ of (k,l)-*pre-seeds *is the smallest set of tuples (s, σ, P, Q), where P and Q are lists of positions in $\mathcal{P}os(s\sigma{\downarrow_{S_E}})$ of L-redexes and G-redexes respectively, such that*

(i) $p\mathcal{S}_l^k$ contains the trivial pre-seed $(x, \{x \mapsto L\}, \{\Lambda\}, \{\,\})$ *;*

(ii) $p\mathcal{S}_l^k$ is closed by nested overlapping*: given $(s, \sigma, P, Q) \in p\mathcal{S}_l^k$, two lists of parallel positions $\{p_i \in \mathcal{FP}os(s\sigma{\downarrow_{S_E}}) : p_i \geq_{\mathcal{P}} P \cdot F_L\}_{i\in I}$ and $\{q_j \in \mathcal{FP}os(s\sigma{\downarrow_{S_E}}) : q_j \geq_{\mathcal{P}} Q \cdot F_G\}_{j\in J}$ which are not both empty and whose elements are pairwise incomparable, renamings $\{L_i\}_{i\in I}$ of L and $\{G_j\}_{j\in J}$ of G such that $\forall i, j$, $\mathcal{V}ar(L_i)$, $\mathcal{V}ar(G_j)$ and $\mathcal{V}ar(s\sigma)$ are pairwise disjoint sets, and τ a most general SE-unifier of the equation $\bigwedge_{i\in I}(s\sigma{\downarrow_{S_E}})|_{p_i} = L_i \wedge \bigwedge_{j\in J}(s\sigma{\downarrow_{S_E}})|_{q_j} = G_j$, the* non-trivial pre-seed $(s\sigma{\downarrow_{S_E}}, \tau, P \cup \{p_i\}_{i\in I}, Q \cup \{q_j\}_{j\in J})$ *belongs to $p\mathcal{S}_l^k$.*

The set of seeds *is defined as $\mathcal{S}_l^k \stackrel{\text{def}}{=} \{(u\tau{\downarrow_{S_E}}, P, Q) : (u, \tau, P, Q)$ is a non-trivial pre-seed$\}$.*

In this recursive definition of pre-seeds, the last overlapping substitution τ tells us where to overlap next, while the maximal positions in P and Q of these overlaps for both rules tell us where to not overlap. In particular, the very first overlap is impossible with L, unless $k = l$: our notation $\mathcal{S}_l^k$ is meant to suggest that there is a k-redex above the l-redexes. The fact that overlapping at some recursive call may take place inside the substitution obtained at the previous recursive call explains the need of a recursive computation of the pre-seeds.

Note that, for the first nested overlapping computation, the set $\{p_i\}_{i\in I}$ is empty, so that the first (parallel) overlap is always with the sole rule $G \to D$ overlapping L at the set of parallel positions $\{q_j\}_j$.

Since the sets P, Q are monotonically increasing (wrt to the order on positions), our computation of pre-seeds avoids too many redundancies. Filtering out the remaining redundancies would require some more bookeeping.

One may wonder why we call these critical peaks nested rather than orthogonal. First, orthogonality refers explicitly to the absence of critical pairs, so an orthogonal critical pair would be kind of self-contradicting. Another reason is that there is a single rule lefthand side sitting at the top of a seed. Therefore, all redexes occurring in a seed are nested inside that lefthand side, whether they extend the seed construction vertically or horizontally.

There is no reason why the set of pre-seeds, hence the set of seeds, should be finite. In case it is finite, then confluence is checkable. Otherwise, confluence needs to be proved. Here is such an example:

Example 3.2. *Consider the rules $1 : f(f(x)) \to x$ and $2 : f(y) \to y$, and let us compute first the pre-seeds between rule 1 and 2*

Init: $(z, \{z \mapsto f(f(z))\}, \{\Lambda\}, \{\,\})$ is a seed.

Step 1: the set $\{p \in \{\Lambda, 1\} : p >_{\mathcal{P}} \{\Lambda\} \cdot \{\Lambda, 1\} = \{\,\}$. The set $\{q \in \{\Lambda, 1\} : q >_{\mathcal{P}} \{\,\} \cdot \{\Lambda\} = \{\Lambda, 1\}$ and we choose the list made of the single position Λ. The unifier of the equation $f(f(z)) = f(y)$ yields $\{y \mapsto f(z)\}$. Therefore

$(f(f(z)), \{y \mapsto f(z)\}, \{\Lambda\}, \{\Lambda\})$ is a seed.

Step 3: the first set remains the same while the second becomes $\{1\}$. We can then choose the list made of that position. The unifier of the equation $f(z) = f(y)$ yields $\{y \mapsto z\}$. Therefore

$(f(f(z)), \{y \mapsto z\}, \{\Lambda\}, \{\Lambda, 1\})$ is a seed.

Step 4: the computation stops with empty sets of p's and q's.

Since this computation is non-deterministic in the choice of positions, we could actually have done step 3 directly, instead of step 2. This would have given the very same seed as in step 3 (because the susbtitution obtained at step 2 had no effect on the overlap term).

Let us now compute the pre-seeds between rule 2 and rule 1.

Init: $(z, \{z \mapsto f(z)\}, \{\Lambda\}, \{\,\})$ is a pre-seed.

Step 1: The sets of p's and q's are respectively $\{\,\}$ and $\{\Lambda\}$. Choosing $q = \Lambda$, we get $\tau = \{z \mapsto f(x)\}$, hence, renaming x in z, we get $(f(z), \{z \mapsto f(f(z))\}, \{\Lambda\}, \{\Lambda\})$ is a pre-seed.

Step 2 yields the pre-seed $(f(f(z)), \{\,\}, \{\Lambda, 1\}, \{\Lambda\})$, and the comptation stops at step 3 with no further seed.

Computing now the pre-seeds between rule 1 and itself, we get the single pre-seed $(f(z), \{\,\}, \{\Lambda\}, \{\Lambda\})$.

We can now compute the pre-seeds between rule 2 and itself and get infinitely many elements of one of the following two sets: $\{(\{f^{2n}(z), \{z \mapsto f(z)\}, \{1^i\}_{i=0}^{i=2n}, \{1^i\}_{i=1}^{i=2n-1})\}_{n>0}$ and $\{(\{f^{2n+1}(z), \{z \mapsto f(z)\}, \{1^i\}_{i=0}^{i=2n-1}, \{1^i\}_{i=1}^{i=2n})\}_{n>0}$.

A straightforward induction on the definition of pre-seeds shows:

Lemma 3.10. *Let $(u, P, Q) \in \mathrm{S}_l^k$ be a seed. Then, $u = L_k\varphi{\downarrow_{S_E}}$ for some φ and $v \overset{P}{\underset{k}{\Longleftarrow\!\!\otimes}} u \overset{Q}{\underset{l}{\otimes\!\!\Longrightarrow}} w$.*

Definition 3.11. *Let $(u, P, Q) \in \mathcal{S}_l^k$ and θ an arbitrary substitution. Then, $v \overset{P}{\underset{k}{\Longleftarrow\!\!\otimes}} u \overset{Q}{\underset{l}{\otimes\!\!\Longrightarrow}} w$, the tuple $((u\theta){\downarrow_{S_E}}, k, P, l, Q)$, and $(v\theta){\downarrow_{S_E}} \overset{P}{\underset{k}{\Longleftarrow\!\!\otimes}} (u\theta){\downarrow_{S_E}} \overset{Q}{\underset{l}{\otimes\!\!\Longrightarrow}} (w\theta){\downarrow_{S_E}}$, are respectively a* nested critical peak *a* nested overlap *; and a* nested peak*.*

Note that Lemma 2.12 is implicitly used in these definitions. Note also that the quadruple (k, P, l, Q) defines already a family of nested overlaps, since the terms $u = L_k\sigma\theta{\downarrow_{S_E}}$ can be recomputed from it, σ being a most general SE-unifier among possibly many that satisfy the corresponding set of equations given in the definition of the set of pre-seeds. In case θ is the identity, we then get a nested *critical* overlap.

We now give a useful characterization of nested overlaps.

Lemma 3.12. *Assume $(u, k, P = \{p_i\}_{i\in I}, l, Q = \{q_j\}_{j\in J})$ is a nested overlap. Then, $\bigwedge_{i\in I, p\in P} u|_p =_{SE} L_i\sigma_i$ and $\bigwedge_{j\in J, q\in Q} u|_q =_{SE} G_j\tau_j$, where $\{L_i\}_{i\in I}$ and $\{G_i\}_{j\in J}$ are variable renamings of L, G sharing no variables among themselves nor with u.*

Proof. We assume implicitely here that the positions in P, Q are listed in non-decreasing order wrt $>_{\mathcal{P}}$. The proof is by induction on $P \cup Q$. Let $O = max(P)$, $O' = max(Q)$, $P' = P \setminus O$ and $Q' = Q \setminus O'$. It follows from the definition of nested overlaps that $(u, k, P' = \{p_i\}_{i\in I'}, l, Q' = \{q_j\}_{j\in J'})$ is a nested overlap. By induction hypothesis, it satisfies the property $\bigwedge_{i\in I', p\in P'} u|_p =_{SE} L_i\sigma_i$ and $\bigwedge_{j\in J', q\in Q'} u|_q =_{SE} G_j\tau_j$, where $\{L_i\}_{i\in I}$ and $\{G_i\}_{j\in J}$ are variable renamings of L, G sharing no variables among them-

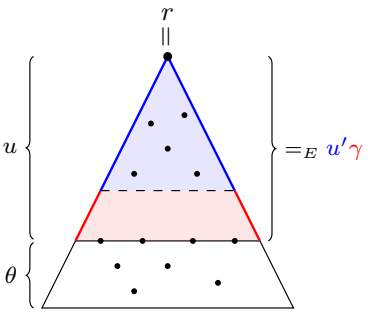

Figure 2. Critical peak property of orthogonal rewriting

selves nor with u. We conclude by using the fact that u rewrites in parallel at O with rule i and at O' with rule j. □

We conclude with the critical peak property whose main idea is represented at Figure 2. Coulours intend to figure out the application of equational E-steps on the term u.

Theorem 3.1 (Critical pairs)**.** *Let* $s \underset{i:L\to R}{\overset{P}{\Longleftarrow\!\!\otimes}} r \underset{j:G\to D}{\overset{Q}{\otimes\!\!\Longrightarrow}} t$, $\Lambda \in P$ *and* $Q_{min} \cap \mathcal{FP}os(L) \neq \emptyset$. *Then,* $\exists u, v, w, u', v', w', \theta, \sigma, \tau, \gamma, O, O', P', Q'$ *st:*

(i) $r = u[\theta], s = v[\sigma], t = w[\tau]$, *and* $u =_E u'\gamma\downarrow_{S_E}, v =_E v'\gamma\downarrow_{S_E}, w =_E w'\gamma\downarrow_{S_E}$,

(ii) $P = O \otimes_u P'$ *and* $Q = O' \otimes_u Q'$,

(iii) $v' \underset{i}{\overset{O}{\Longleftarrow\!\!\otimes}} u' \underset{j}{\overset{O'}{\otimes\!\!\Longrightarrow}} w'$ *is a nested critical peak,*

(iv) and θ *is a preserving substitution st* $\sigma \underset{i}{\overset{P'}{\Longleftarrow\!\!\otimes}} \theta \underset{j}{\overset{Q'}{\otimes\!\!\Longrightarrow}} \tau$.

The substitution θ above is the one obtained by using Lemma 3.8 to split r as $u[\theta]$ so that u contains a critical overlap. The substitution γ is then obtained by expressing the property that u is an instance of the most general critical peak (v', u', w'), hence $u =_E u'\gamma$. These two substitutions play very different roles, it is important to keep them separate. In particular, the substitution γ is not preserving in general, but does not rewrite. On the other hand, the substitution θ is preserving and rewrites.

Example 3.3. *Consider the plain rewrite system* $\{f(f(x)) \to x\}$. *Let* $s \overset{\text{def}}{=} f(f(f(a))), P \overset{\text{def}}{=} \{\Lambda, 11\}, r \overset{\text{def}}{=} f(f(f(f(f(f(f(a))))))),$ $t \overset{\text{def}}{=} f(f(f(a))), Q \overset{\text{def}}{=} \{1, 11111\}$. *Then,* $s \underset{}{\overset{P}{\Longleftarrow\!\!\otimes}} r \overset{\text{def}}{=} \underset{}{\overset{Q}{\otimes\!\!\Longrightarrow}} t$. *Let now* $u \overset{\text{def}}{=} f(f(f(f(f(z))))), O \overset{\text{def}}{=} \{\Lambda, 11\}, v \overset{\text{def}}{=} f(z), O' \overset{\text{def}}{=} \{1\}, w \overset{\text{def}}{=} f(f(f(z)))$. *Then ,* $v \underset{}{\overset{O}{\Longleftarrow\!\!\otimes}} u \overset{\text{def}}{=} \underset{}{\overset{O'}{\otimes\!\!\Longrightarrow}} w$. *Let now* $\theta \overset{\text{def}}{=} \{z \mapsto f(f(a))\}, P'_z \overset{\text{def}}{=} \{\}, \sigma \overset{\text{def}}{=} \{z \mapsto f(f(a))\}, Q'_z \overset{\text{def}}{=} \{111\}, \tau \overset{\text{def}}{=} \{z \mapsto a\}$. *Then* $z\sigma \underset{}{\overset{P'}{\Longleftarrow\!\!\otimes}} z\theta \underset{}{\overset{Q'}{\otimes\!\!\Longrightarrow}} z\tau$.

Note that the absence of an equational theory E *and the linearity of the lefthand side of the rule* $f(f(x)) \to x$ *implies that* γ *is the identity. Then,* $u' = u, v' = v, w' = w$.

Proof. Since $\Lambda \in P$, r must be S_E normal by definition of higher-order rewriting, as well as its subterms. By assumption on P, Q, the two rewrites from r overlap. Let O, O' be the maximal *prefixes* of P, Q, (that is, $\forall p \in P(p \leq_{\mathcal{P}} P \Rightarrow p \in O)$ and $\forall q \in Q(q \leq_{\mathcal{P}} Q \Rightarrow q \in Q)$), such that $(i, O = \{O_i\}_{i\in I}, j, O' = \{O'_j\}_{j\in J})$ defines a nested overlap (r, i, O, j, O') whose associated nested peak is $s' \underset{i}{\overset{O}{\Longleftarrow\!\!\otimes}} r \underset{j}{\overset{O'}{\otimes\!\!\Longrightarrow}} t'$ for some s', t'.

By the construction of pre-seeds, $O \cup O'$ is a non-empty prefix of $P \cup Q$ (this is of course not true of arbitrary prefixes of P, Q). Let then $N \overset{\text{def}}{=} ((P \cup Q)\backslash(O \cup O'))_{min}$ and $u \overset{\text{def}}{=} r[\overline{z}]_N$, hence $r = u[\theta]$ for some θ normal of domain $\overline{z}$ such that $z\theta$ is either an i-redex or a j-redex, hence is headed by a symbol from Σ_{ho}. The substitution θ is therefore preserving. By Lemma 3.3, $P = O \otimes_u P'$ and $Q = O' \otimes_u Q'$. By definition of orthogonal rewriting and maximality of O, O', the positions $n \in N$ do not belong to any set of the form $p \cdot \{o <_{\mathcal{P}} F_L \cup F_G\}$ unless $p = n$ or $q \cdot \{o <_{\mathcal{P}} F_L \cup F_G\}$ unless $q = n$. Hence $u[\theta] \underset{i}{\overset{O\otimes_r P'}{\otimes\!\!\Longrightarrow}} v[\sigma]$ for some σ and $u[\theta] \underset{j}{\overset{O'\otimes_r Q'}{\otimes\!\!\Longrightarrow}} w[\tau]$ for some τ by Lemma 3.8 applied twice. Therefore $s = v[\sigma]$ and $t = v[\tau]$ by Corollary 3.5 applied twice.

We have got the peaks $v \underset{i}{\overset{O}{\Longleftarrow\!\!\otimes}} u \underset{j}{\overset{O'}{\otimes\!\!\Longrightarrow}} w$ and $\sigma \underset{i}{\overset{P'}{\Longleftarrow\!\!\otimes}} \theta \underset{j}{\overset{Q'}{\otimes\!\!\Longrightarrow}} \tau$. By Lemma 3.12, $\bigwedge_{i\in I, p\in O} u|_p =_{SE} L_i\sigma_i$ and $\bigwedge_{j\in J, q\in O'} u|_q =_{SE} G_j\tau_j$, where $\{L_i\}_{i\in I}$ and $\{G_i\}_{j\in J}$ are variable renamings of L, G sharing no variables among themselves nor with u. Hence $\bigwedge_{i\in I, p\in O, j\in J, q\in O'} u|_p =_{SE} L_i\sigma_i \wedge u|_q =_{SE} G_j\tau_j$ holds. Thanks to the conditions on variables of L_i, G_j, the corresponding pre-seeds unification problem is therefore unifiable modulo SE. By property of most general SE-unifiers, $\sigma_i =_{SE} \varphi\gamma$ and $\tau_j =_{SE} \varphi\gamma$ for some substitutions φ of domain $\bigcup_{i\in I} \mathcal{D}om(\sigma_i) \cup \bigcup_{j\in J} \mathcal{D}om(\tau_j)$ and γ. Note that we use the fact that the substitutions from the set $\{\sigma_i, \tau_j\}_{i\in I, j\in J}$ have pairwise disjoint domains.

Let $u' = L\varphi\downarrow_{S_E}$. By Lemma 3.10, $v' \underset{i}{\overset{O}{\Longleftarrow\!\!\otimes}} u' \underset{j}{\overset{O'}{\otimes\!\!\Longrightarrow}} w'$. By the Church-Rosser property of (S, E), $u =_E u'\gamma\downarrow_{S_E}$. Hence $v =_E v'\gamma\downarrow$ and $w =_E w'\gamma\downarrow$ by Corollary 3.5 and the Church-Rosser property of (S, E). This ends the proof. □

4. Decreasing diagrams for labelled rewriting

Our goal is to prove confluence on untyped terms. Since beta-reductions do not terminate on their set, we shall use van Oostrom's technique relying on the existence of a decreasing diagram for each local peak [28]. Since functional rewrites operate modulo variable renaming, we shall also use Jouannaud and Liu's extension of van Oostrom's decreasing diagram completeness property, for which commutation plays a central role [18].

Labelled rewriting. Labelled rewriting is a basic notion underlying decreasing diagrams. A labelled binary relation on an abstract set is denoted by $u \xrightarrow{m} v$, where m is an element of a set $\mathcal{L}$ of *labels* equipped with a partial quasi-order $\trianglerighteq$ which strict part $\triangleright$ is well-founded. We write $m = n$ (resp., $m \# n$) for equivalent (resp., incomparable) labels m, n, and $\alpha \triangleright l$ (resp., $l \triangleright \alpha$) if $m \triangleright l$ (resp., $l \triangleright m$) for all m in the multiset (or sequence) α of labels.

Decreasing diagrams [30].

Definition 4.1 (Local diagram)**.** *Given a labelled relation* $\longrightarrow$ *on an abstract set, a* local diagram D *is a pair made of a* local peak $D_{peak} \overset{\text{def}}{=} v \longleftarrow u \longrightarrow w$ *and a* conversion $D_{conv} \overset{\text{def}}{=} v \longleftrightarrow\!\!\!\!\!\!\rightarrow w$.

Definition 4.2 (Decreasing local diagram)**.** *A local diagram with peak* $v \xleftarrow{m} u \xrightarrow{n} w$ *is* decreasing *if its* joining conversion *has the form* $v \overset{\alpha}{\longleftrightarrow\!\!\!\!\!\!\rightarrow} s \xrightarrow{n} s' \overset{\gamma}{\longleftrightarrow\!\!\!\!\!\!\rightarrow} t' \xleftarrow{m} t \overset{\beta}{\longleftrightarrow\!\!\!\!\!\!\rightarrow} w$, *with labels in* α *(resp.* β, *resp.* γ*) strictly smaller than* m *(resp.* n, *resp.* m *or* n*).* $s \xrightarrow{n} s'$ *and* $t' \xleftarrow{m} t$ *(resp.,* $v \xrightarrow{\alpha} s$, $t \xleftarrow{\beta} w$, *resp.,* $s' \overset{\gamma}{\longleftrightarrow\!\!\!\!\!\!\rightarrow} t'$*) are called the* facing *(resp.* side*, resp.* middle*) steps of the conversion.*

Decreasing diagrams are abbreviated as DDs. A facing step of a decreasing diagram may be missing, its side steps are then absorbed by the middle ones. This is the case if $m \rhd n$ (or $n \rhd m$).

Theorem 4.1 ([30])**.** *A labelled, abstract rewriting relation is Church-Rosser if all its local peaks have a decreasing diagram. Conversely, any confluent relation on a countable set can be labelled so that all its local peaks have decreasing diagrams.*

The modulo case. van Oostrom's theorem generalizes to rewrite relations modulo an equational theory. Its converse requires two further properties: that rewriting commutes over the equational theory and that the set of labels is increased by a new minimum label reserved for the equational steps [18]. This generalization of van Oostrom's theorem will be applied later to $\Longrightarrow_{\beta_\alpha}$.

5. Confluence in $\lambda\Pi$Mod

We now consider the Church-Rosser property of the rewrite relation used in $\lambda\Pi$Mod on untyped terms, generated by the rewriting system whose pieces have been described in Section 2. As usual, it is essential to choose carefully the relation to work with. For the method to be sound, it must contain rewriting and define the same convertibility relation as the one generated by the set of rules/equations. In case of non-terminating rewrites, non-linearities make it difficult to get decreasing diagrams for ancestor peaks since there can only be one facing step on each side of the joining conversion. The usual way out is to impose left-linearity (for non-confined variables is actually enough) and use some form of parallel rewriting to handle non-right-linearities. First-order rewrites having a small label will not need parallel rewriting, but β-rewrites will in case they take place below a first-order rewrite. Higher-order rewrite steps will require a label bigger than the other steps, so that the latters can be neglected when needed. Parallel higher-order rewriting will be needed again, but is no more sufficient: parallel higher-order rewrites taking place below a beta-step may now become both duplicated and nested, making orthogonal rewriting necessary to get decreasing diagrams. The various rewrite steps to be considered in an arbitrary conversion, from which all local peaks and cliffs must be replaced by decreasing diagrams, are therefore:

$$\underset{(R_{fo},S_{cd},E_{cd})}{\longrightarrow} \cup \underset{E_{cd}}{\longleftrightarrow} \cup \underset{\beta}{\Longrightarrow} \cup =_\alpha \cup \underset{(R_{ho},S,E)}{\otimes\!\!\Longrightarrow}$$

We shall also use when needed the notation $\longrightarrow$ as an abbreviation for $\underset{(R_{fo},S_{cd},E_{cd})}{\longrightarrow} \cup \underset{\beta^0}{\longrightarrow} \cup \underset{(\beta^{\neq 0},\alpha)}{\Longrightarrow} \cup \underset{(R_{ho},S,E)}{\otimes\!\!\Longrightarrow}$.

5.1 Labelling rewrites

We label each rewrite or equational step by a pair $\langle m, n, p\rangle$, where m is a natural number indicating the rewrite category, n is a natural number or a term, and p is the rewrite position or set of positions, which is used here for convenience, not for comparing labels. Labels do not depend upon the direction of a rewrite step. Labels are compared lexicographically, using the order on natural numbers for all their components but terms, which are compared in the order defined by the termination property of (R_{fo}, S_{cd}, E_{cd}).

Rewrite	m	n
$u =_\alpha v$	0	0
$u \longleftrightarrow^p_{E_{cd}} v$	0	1
$u \longrightarrow^p_{(R_{fo},S_{cd},E_{cd})} v$	1	u
$u \Longrightarrow^P_\beta v$	2	// canonical labelling
$u \otimes\!\!\Longrightarrow^Q_{i\in R_{ho}} v$	3	i

There are three different categories of rewrite steps: first-order, functional and higher-order, and two kinds of equational steps: first-order and functional. The label's structure obeys two important principles. First, only the second label matters for *homogeneous* local peaks, which use a rewrite of the same category on both sides of the peak. On the other hand, finding decreasing diagrams for *heterogeneous* local peaks relies on the first label's component.

Equalities in E_{cd} and α-conversion are treated very differently. While the later is considered as an equational theory for which all rewrites commute over, $\longleftrightarrow_{E_{cd}}$ is considered instead as a symmetric rewrite relation whose local peaks with the other three rewrite relations, called *cliffs*, must have decreasing diagrams like the other local peaks. This subtlety eases the use of the completeness theorem for decreasing diagrams modulo, for which $=_\alpha$-steps must have the same label, minimum among all labels [18].

5.2 Decreasing reductions

So far, we have used essentially preserving substitutions replacing subterms at variable positions, since this is the essential notion needed in the definition of orthogonal rewriting. However, Theorem 3.1 introduces substitutions that are not preserving. Their use requires a specific treatment carried out now.

Definition 5.1. *Given a normal term u, a rewrite step $u \xrightarrow{\Lambda} v$ is* collapsing *if v is a non-confined variable or an abstraction.*

The following property follows directly from the definition:

Lemma 5.2. *Non-collapsing rewrites preserve β^0-normal forms.*

As a preparation for the study of critical higher-order peaks, we define below in two steps, reductions which are intended to be decreasing for all instantiations.

Definition 5.3. *A derivation is called* eager-normal *if of the form $u \underset{S_E}{\longrightarrow\!\!\!\!\!\rightarrow} u{\downarrow_{S_E}} =_E s$ or $u \underset{S_E}{\longrightarrow\!\!\!\!\!\rightarrow} u{\downarrow_{S_E}} =_E s \underset{i}{\longrightarrow} t \longrightarrow\!\!\!\!\!\rightarrow v$ where $s \underset{i}{\longrightarrow} t$ is called an* essential *step, and $t \longrightarrow\!\!\!\!\!\rightarrow v$ is an eager-normal derivation.*

Note that the essential step cannot be a simplification step, since (S, E) is Church-Rosser and terminating, and $u{\downarrow_{S_E}}$ is normal.

Lemma 5.4. *Let u be a normal term st $u \longrightarrow\!\!\!\!\!\rightarrow v$ is an eager-normal derivation whose essential steps are non-collapsing. Then, given σ, there is an eager-normal derivation from $u\sigma$ to $v\sigma$ whose essential steps use the same rules and are non-collapsing.*

Note that there is no need to normalize $u\sigma$ and $v\sigma$ here, since this is done anyway in the definition of an eager-normal derivation.

Note also that the assumption that essential steps are non-collapsing is crucial: otherwise, instantiation of variables by abstractions could create new β^0-redexes.

Proof. By induction on the sequence, assuming wlog that σ is normal. Let $u \underset{S_E}{\longrightarrow\!\!\!\!\!\rightarrow} u{\downarrow_{S_E}} =_E s$ be the S_E-normalization prefix of the reduction. Since (S, E) is Church-Rosser, then $u\sigma{\downarrow_{S_E}} =_E s\sigma{\downarrow_{S_E}}$. If $s = v$, we are done. Otherwise, $s \underset{i}{\xrightarrow{p}} t$ with $i \in \{\beta\} \cup R_{fo} \cup R_{ho}$, and $t \longrightarrow\!\!\!\!\!\rightarrow v$ is eager-normal. First, $s\sigma{\downarrow_{S_E}} = (s\sigma[s\sigma|_p]_p){\downarrow_{S_E}} = (s\sigma[s|_p\sigma]_p){\downarrow_{S_E}} = (s\sigma[(s|_p\sigma){\downarrow_{S_E}}]_p){\downarrow_{S_E}}$. There are three cases:

- $i = \beta$. Then $s|_p = (\lambda x.w\ w')$ and $t = s[w\{x \mapsto w'\}]_p$, where w, w' are normal and $s|_p$ is not a β^0-redex. Now, $s|_p\sigma{\downarrow_{S_E}} = (\lambda x.w\sigma{\downarrow_{S_E}}\ \ w'\sigma{\downarrow_{S_E}}) \xrightarrow[\beta]{\Lambda} w\sigma{\downarrow_{S_E}}\ \{x \mapsto w'\sigma{\downarrow_{S_E}}\}$. Hence $s\sigma{\downarrow_{S_E}} \xrightarrow[\beta]{p} (s\sigma[w\sigma{\downarrow_{S_E}}\ \{x \mapsto w'\sigma{\downarrow_{S_E}}\}]_p){\downarrow_{S_E}} \underset{S_E}{\longrightarrow\!\!\!\!\!\rightarrow} (s\sigma[w\sigma\{x \mapsto w'\sigma\}]_p){\downarrow_{S_E}} =_E t\sigma{\downarrow_{S_E}}$.

- $i : l \to r \in R_{fo}$. Then, $s|_p \underset{i}{\longrightarrow} t|_p$. Since σ is normal and rule i is first-order, $(s|_p\sigma)\downarrow_{S_E} = (s|_p\sigma)\downarrow_{S_{cd}E_{cd}} \underset{i}{\longrightarrow} (t|_p\sigma)\downarrow_{S_{cd}E_{cd}} = (t|_p\sigma)\downarrow_{S_E}$. Hence $s\sigma\downarrow_{S_E} \underset{i}{\overset{p}{\longrightarrow}} (t\sigma[(t|_p\sigma)\downarrow_{S_E}]_p)\downarrow_{S_E} = t\sigma\downarrow_{S_E}$.
- $i \in R_{ho}$. Follows from Lemma 2.12.

In all cases, we proceed by induction on $t \longrightarrow\!\!\!\!\!\rightarrow v$ to obtain the announced eager-normal reduction ending in $v\sigma\downarrow$. □

These results extend straightforwardly to eager-normal reductions having orthogonal steps, by expanding them first, and then folding them again.

Definition 5.5. *Given a local peak (v, u, w) labelled $\langle m \in \{1, 3\}, i, P\rangle$ and $\langle 3, j, Q\rangle$ from u to v and u to w respectively, a* decreasing reduction *is a derivation from some term s st*

(i) $s \in \{v, w\}$ and all steps have a label strictly smaller than either $\langle m, i\rangle$ or $\langle 3, j\rangle$; or

(ii) $m = 3$ and $s = v \cdots \underset{k}{\overset{\langle l,q\rangle}{\longrightarrow}} \cdots \underset{j}{\overset{\langle 3,j\rangle}{\Longrightarrow\!\!\!\!\!\rightarrow}} \cdots \underset{k'}{\overset{\langle l',q'\rangle}{\longrightarrow}} \cdots$ with $l \lhd \langle 3, i\rangle$ and $l' \lhd \langle 3, max(i, j)\rangle$ (the case $s = w$ is symmetrical).

A non-collapsing eager-normal decreasing reduction is called strongly decreasing *(SDR).*

The main restriction of an SDR is that the first form applies when $m = 1$ and $s = v$, and therefore no facing step is allowed in this case. This is because normalization β^0-steps would otherwise be necessary *before* the facing step, which would contradict van Oostrom's condition for decreasing diagrams. Although this restriction is satisfied by our example, it is not very satisfactory, and believe it is non-necessary. The reason is that, given a higher-order step at position p, it is possible to hide the β^0-steps taking place before at positions below p, by including them to the definition of the higher-order rewrite at p. We have not tried this idea yet because our example satisfies the restriction.

Being an eager-normal decreasing reduction is of course no real restriction, but the absence of collapsing essential steps is. However, since higher-order rewrites are non-collapsing by Assumption 6 (i,ii), the only further restriction is that β-steps and first-order steps are non-collapsing. And since *typed* first-order rewrites cannot pop-up abstractions, only collapsing β-steps are indeed forbidden, a quite weak restriction for practice. This restriction is satisfied as well by our example.

Example 5.1. *An example of SDR which is not a develomment.*

Given the previous local peak (v, u, w), we use the notation $\rightrightarrows\!\!<\!m, i; n, j\!>\!\!\rightrightarrows$ for SDRs originating from v, and $\leftrightarrows\!\!<\!m, i; n, j\!>\!\!\leftrightarrows$ for those originating from w. We sometimes omit unnecessary components of these 4-tuples.

As a consequence of Lemma 5.4, we get:

Corollary 5.6 (Stability of SDRs)**.** *Let $u \rightrightarrows\!\!<\!m, i; n, j\!>\!\!\rightrightarrows w$ be an SDR. Given σ, there is an SDR $u\sigma \rightrightarrows\!\!<\!m, i; n, j\!>\!\!\rightrightarrows w\sigma$.*

An important property for constructing decreasing diagrams for higher-order local peaks is the ability to permute steps taking place in the term u and the capturing substitution σ of a term $u[\sigma]$ so as to bring together distant orthogonal i-steps taking place respectively in u and σ and merge them in a single orthogonal i-step taking place in $u[\sigma]$. This is possible for strongly decreasing reductions, thanks again to the non-collapsing assumption (for the essential steps behind the facing higher-order step would suffice here).

Lemma 5.7. *Let σ preserving such that $s[\sigma] \underset{i}{\longrightarrow} t[\sigma] \underset{j}{\longrightarrow} t[\tau]$. Then, $s[\sigma] \underset{j}{\longrightarrow} s[\tau] \underset{S_E}{\longrightarrow\!\!\!\!\!\rightarrow} \underset{i}{\longrightarrow} \underset{S_E}{\longrightarrow\!\!\!\!\!\rightarrow} t[\tau]\downarrow_{S_E}$.*

Proof. The case where j a higher-order rewrite follows from Corollary 3.7. If j is a β-rewrite or a first-order rewrite and i is a higher-order rewrite, the use of Lemma 2.12 introduces extra rewrites steps with S_E. Other cases are straightforward. □

Lemma 5.8. *Let s be a normal term and σ is a preserving substitution such that $s \rightrightarrows\!\!<\!k, j; 3, i\!>\!\!\rightrightarrows u$ and $\sigma \rightrightarrows\!\!<\!k, j; 3, i\!>\!\!\rightrightarrows \theta$ are two SDRs. Then, there is an SDR $s[\sigma] \rightrightarrows\!\!<\!k, j; 3, i\!>\!\!\rightrightarrows u[\theta]$.*

Proof. By definition of an SDR, $s[\sigma] \overset{\lhd\langle k,j\rangle}{\longrightarrow\!\!\!\!\!\rightarrow} s_1[\sigma] \underset{\langle 3,i\rangle}{\otimes\!\!\Longrightarrow} s_2[\sigma]$ $\overset{\lhd\langle 3,i\rangle,\langle k,j\rangle}{\longrightarrow\!\!\!\!\!\rightarrow} u[\sigma] \overset{\lhd\langle k,j\rangle}{\longrightarrow\!\!\!\!\!\rightarrow} u[\sigma_1] \underset{\langle 3,i\rangle}{\otimes\!\!\Longrightarrow} u[\sigma_2] \overset{\lhd\langle 3,i\rangle,\langle k,j\rangle}{\longrightarrow\!\!\!\!\!\rightarrow} u[\theta]$.

We shall recursively transform this derivation until the subderivation between the two orthogonal steps is empty, in which case they can be merged by definition of orthogonal rewrites. The induction is on the pair (m, n) where m is the number of $\lhd i, j$-steps between both orthogonal i-steps, and m is the number of contiguous $\lhd j$-steps to the left of the rightmost orthogonal i-step. There are four cases:

1. $m = n = 0$. This is the basic case of the induction. The result follows by merging both orthogonal steps thanks to the definition of orthogonal rewrites.
2. $m = 0$ and $n \neq 0$. Then, we move the leftmost orthogonal i-step to the right thanks to Lemma 5.7, which decreases n by one and leaves m unchanged.
3. $m \neq 0$ and $n = 0$. Then, we move the rightmost $\lhd i, j$-step over the rightmost orthogonal i-step by using Lemma 5.7, which decreases m by one.
4. $m, n \neq 0$. Then, we move the rightmost $\lhd i, j$-step over the following $\lhd j$-step by using Lemma 5.7, which decreases n by one and leaves m unchanged.

In all cases but the first, we conclude by induction hypothesis. □

In the sequel, we call *strongly decreasing diagrams* (SDDs) those that use strongly decreasing reductions on both sides.

5.3 DDs for first-order local peaks and cliffs

Decreasing diagrams result from our assumption that the equational rewrite system (R_{fo}, S_{cd}, E_{cd}) is Church-Rosser and terminating:

Corollary 5.9. *Homogeneous first-order local peaks and cliffs have decreasing diagrams.*

There is a subtlety in the proof. Consider a local peak $s \underset{i}{\overset{p}{\longleftarrow}} u \underset{j}{\overset{q}{\longrightarrow}} t$. By the Church-Rosser assumption, $s \longrightarrow\!\!\!\!\!\rightarrow v =_{E_{cd}} w \longleftarrow\!\!\!\!\!\!\!\!\rightarrow t$. Every term $r \in \{s, \ldots, v, w, \ldots, t\}$ is strictly smaller than u. Since the labels of the peak are both equal to $\langle 1, u\rangle$, it does not matter whether the first component of the label of the step originating from r is 1 or 0. And since one of these two labels suffices to make the diagram decreasing, the argument applies as well if $i \in R_{fo}, j \in S_{cd}$ or $i \in R_{fo}, j \in E_{cd}$. If $i \in S_{cd}$ and $j \in S_{cd} \cup E_{cd}$, then u must be confined. Then, all rewrites must be confined and hence have 0 as their label's first component.

5.4 DDs for functional local peaks

By parallel canonical labelling, we mean the canonical labelling of the relation $\Longrightarrow_{\beta_\alpha}$, which is Church-Rosser modulo $=_\alpha$ on untyped terms since this is already true of $\longrightarrow_{\beta\alpha}$. We therefore apply the generalisation of van Oostrom's theorem rather than the original version, using the property that $\Longrightarrow_\beta$ commutes over $=_\alpha$.

Note that we need to repeat the same reasonning done previously to account for the fact that the labels of β^0-steps and the other β-steps are different. Again, the key is that the former are smaller.

The fact that they are plain steps rather than parallel steps does not harm because plain steps are particular parallel steps.

Labelling the functional rewrite steps requires further use of an erasing interpretation before to take the parallel canonical labelling, to ensure its invariance by a first-order rewrite taking place below, which will be needed for the analysis of functional/first-order local peaks. Our interpretation is a homomorphism from untyped terms of $\lambda\Pi$Mod to terms of a lambda calculus enriched with a list construction of variable arity denoted by $\{\_\}$. The vector operation over lists is interpreted as concatenation. Lists of lists are flattened. Given a term in which no variable is bound twice,

$$\|(s\ t)\| \stackrel{\text{def}}{=} \{(\|s\|\ \|t\|)\} \qquad \|f(\overline{s})\| \stackrel{\text{def}}{=} \{\overline{\|s\|}\} \qquad \|x \in \mathcal{X}\| \stackrel{\text{def}}{=} \{x\}$$
$$\|\lambda x : t.s\| \stackrel{\text{def}}{=} \{\lambda x : \|t\|.\|s\|\} \qquad \|\Pi x : s.t\| \stackrel{\text{def}}{=} \{\Pi(\|s\|, \|t\|)\}$$

Positions in a list $\{\overline{u}\}$ are as expected, as are the extension of $\lambda\Pi$-constructors from terms to lists of terms.

Being non-erasing homomorphisms by Assumption 4(ii), interpretations are compatible with labelling local functional peaks:

Lemma 5.10. *Assume* $s|_p = (\lambda x.v\ w)$ *for some* $p \in \mathcal{P}os(s)$. *Then,* $\|s\| \xrightarrow[\beta]{q} \|s\|[[\|v\|\{x \mapsto \|w\|\}]]_q$ *for some* $q \in \mathcal{P}os(\|s\|)$.

This justifies the use of the completeness theorem for the lambda-calculus of interpreted terms.

Corollary 5.11. *Homogeneous functional local peaks and cliffs have decreasing diagrams.*

5.5 DDs for functional/first-order peaks and cliffs

Interpretations are compatible with R_{fo}-rewriting as well:

Lemma 5.12. *Let* $s \underset{R_{fo}}{\longrightarrow} t$. *Then,* $\|s\| = \|t\|$.

Proof. Follows from the assumption that non-confined variables in first-order rules appear linearly. Note that algebraic terms, in particular confined ones, are interpreted by the empty list. □

Corollary 5.13. *Heterogeneous functional/first-order local peaks and cliffs have decreasing diagrams.*

Proof. Consider the local peak $v \overset{\langle 2,m,P\rangle}{\underset{\beta}{\longleftarrow}} u \overset{\langle 1,n,p\rangle}{\underset{i}{\longrightarrow}} w$ with $i : l \to r \in R_{fo}$. Since no critical peak is possible here, $P = P_\# \cup Q$ where $P_\# \# p$ is maximal. By (DP) $w \overset{P_\#}{\underset{\beta}{\Longrightarrow}} s \underset{i}{\overset{q}{\longleftarrow}} v'$, with $v' = u$ and $s = w$ if $P_\# = \emptyset$. If $Q = \emptyset$, we are done. Otherwise, there are two cases:

– $P = P_\# \cup \{q\}$ with $p >_\mathcal{P} q$. Since (β) is left-linear, $v \underset{i}{\longrightarrow} t \underset{\beta}{\overset{Q}{\Longleftarrow}} s$ for some t by (LAP). Hence $t \underset{\beta}{\overset{P}{\Longleftarrow}} w$.

– $P = P_\# \cup Q$ with $Q >_\mathcal{P} p$. Since Q is not empty, the β-redexes at Q are below non-confined variables of l, hence linear by Assumption 4(ii). By (LAP), $v \underset{i}{\overset{p}{\longrightarrow}} t \underset{\beta}{\overset{Q'}{\Longleftarrow}} s$ for some t, where $Q' \geq_\mathcal{P} p$, hence $Q' \# P_\#$. Therefore, we obtain $t \underset{\beta}{\overset{P_\# \cup Q'}{\Longleftarrow}} w$.

In both cases, the label of the parallel β-step from w is $\langle 2, m, Q'\rangle$ by Lemma 5.12, resulting in a decreasing diagram from v to w.

The case of cliffs is similar, even simpler: a β-step cannot occur below an equational step since equations in E_{cd} are confined. □

5.6 DDs for higher-order local peaks

Since *typed* first-order rewrites cannot pop-up abstractions, the following assumption is not very different from expected:

Critical peak assumption 1: Higher-order nested critical peaks have strongly decreasing diagrams.

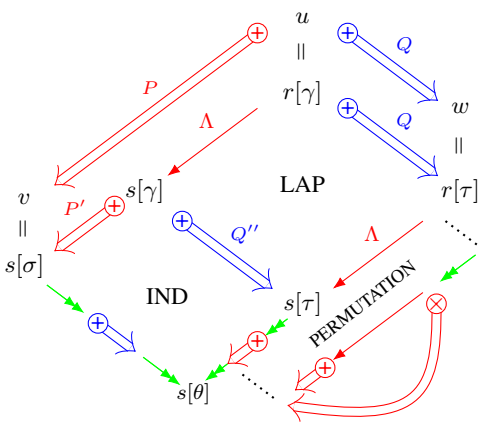

Figure 3. DD for higher-order ancestor peaks

We now show that local peaks $v \overset{\langle 3,i,P\rangle}{\underset{i}{\Longleftarrow\!\!\otimes}} u \overset{\langle 3,j,Q\rangle}{\underset{j}{\otimes\!\!\Longrightarrow}} w$ of orthogonal higher-order rewriting have strongly decreasing diagrams under Critical peak assumption 1, by induction on the size of the term obtained from u by removing its confined subterms.

We consider first the case of a single top step for which $(P \cup Q)_{min} = \{p\}$, assuming wlog that $p = \Lambda \in P$.

Let $O \cup O'$ be the maximal subset of $P \cup Q$ such that (i, O, j, O') defines a nested critical overlap. There are two cases depending upon the possible emptiness of $O \cup O'$.

1. $O = O' = \emptyset$, the non-overlapping case shown at Figure 3.
   Let $r = u[\overline{z}]_{(P\setminus\{\Lambda\}\cup Q)_{min}}$ and $u = r[\gamma]$. By Lemma 3.3, $P = \{\Lambda\} \otimes_r P'$ and $Q = \emptyset \otimes_r Q'$. By Lemma 3.8, we get $v = s[\sigma] \overset{\{\Lambda\}\otimes P'}{\underset{i}{\Longleftarrow\!\!\otimes}} r[\gamma] \overset{\emptyset\otimes Q'}{\underset{j}{\otimes\!\!\Longrightarrow}} r[\tau] = w$. We now decompose the step from u to v into its constituents: $r[\gamma] \overset{\Lambda}{\underset{i}{\otimes\!\!\Longrightarrow}} s[\gamma]$, $\gamma \overset{P'}{\underset{i}{\otimes\!\!\Longrightarrow}} \sigma$. By Corollary 3.7, $s[\gamma] \overset{Q''}{\underset{j}{\otimes\!\!\Longrightarrow}} s[\tau] \underset{i}{\overset{\Lambda}{\longleftarrow}} r[\tau]$. We are done if $P' = \emptyset$, otherwise we get smaller peaks $x\sigma \overset{P'_x}{\underset{i}{\Longleftarrow\!\!\otimes}} x\gamma \overset{Q'_x}{\underset{j}{\otimes\!\!\Longrightarrow}} x\tau$. By induction hypothesis, they have SDRs $x\sigma \rightrightarrows_{<i;\, j>} x\theta \leftleftarrows_{<i;\, j>} x\tau$. By (DP), these SDRs can be amalgamated to form an SDR for the peak $s[\sigma] \underset{i}{\Longleftarrow\!\!\otimes} s[\gamma] \underset{j}{\otimes\!\!\Longrightarrow} s[\tau]$. We finally get an SDR for our starting peak $s[\sigma] \underset{i}{\Longleftarrow\!\!\otimes} r[\gamma] \underset{j}{\otimes\!\!\Longrightarrow} r[\tau]$ by Lemma 5.8.

2. $O \cup O' \neq \emptyset$, the overlapping case shown at Figure 4. By Theorem 3.1, $\exists u', s', t', r, s, t, \theta, \gamma, \sigma, \tau, \varphi, O, O', P', Q'$ st
   (i) $u = u'[\theta]$, $u' =_E r\gamma\downarrow$, $s' =_E s\gamma\downarrow_{S_E}$, $t' =_E t\gamma\downarrow_{S_E}$,
   (ii) $P = O \otimes P'$, $Q = O' \otimes Q'$,
   (iii) $s \overset{O}{\underset{i}{\Longleftarrow\!\!\otimes}} r \overset{O'}{\underset{j}{\otimes\!\!\Longrightarrow}} t$ is a nested critical peak,
   (iv) θ is a preserving substitution such that $\sigma \overset{P'}{\underset{i}{\Longleftarrow\!\!\otimes}} \theta \overset{Q'}{\underset{j}{\otimes\!\!\Longrightarrow}} \tau$.

   By Lemma 3.8, $u = u'[\theta] \overset{O}{\underset{i}{\otimes\!\!\Longrightarrow}} s'[\theta] \overset{P'}{\underset{i}{\otimes\!\!\Longrightarrow}} s'[\sigma] = v$, hence $u' \overset{O}{\underset{i}{\otimes\!\!\Longrightarrow}} s'$ and $\theta \overset{P'}{\underset{i}{\otimes\!\!\Longrightarrow}} \sigma$. Likewise, $u'[\theta] \overset{O'}{\underset{i}{\otimes\!\!\Longrightarrow}} t'[\theta] \overset{Q'}{\underset{j}{\otimes\!\!\Longrightarrow}} t'[\tau] = w$, hence $u' \overset{O'}{\underset{i}{\otimes\!\!\Longrightarrow}} t'$ and $\theta \overset{Q'}{\underset{i}{\otimes\!\!\Longrightarrow}} \tau$.

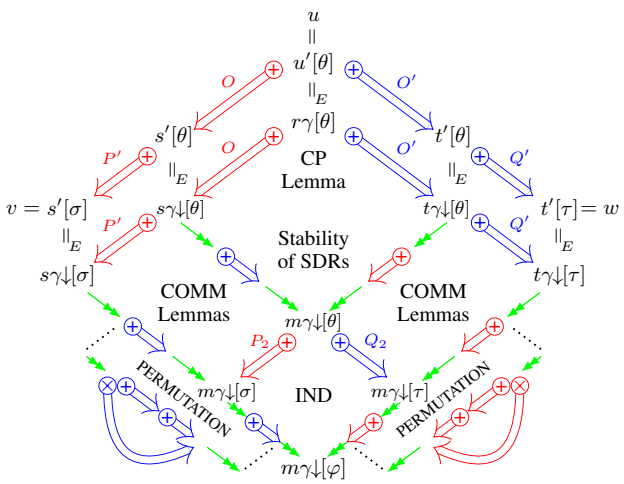

Figure 4. DD for higher-order overlapping peaks

Since u is normal by assumption that $\Lambda \in P$, then u' and θ are normal, hence $r\gamma$ is normal by Lemma 2.9. It follows that r and γ are normal. By Lemma 2.16, s, t are normal. However, $s\gamma, t\gamma$ need not be normal since there is no reason why γ should be preserving. Hence $r\gamma \overset{O}{\underset{i}{\otimes\!\!\Longrightarrow}} s\gamma\!\downarrow_{S_E}$ by Lemma 2.12. Likewise, $r\gamma \overset{O'}{\underset{i}{\otimes\!\!\Longrightarrow}} t\gamma\!\downarrow_{S_E}$. Further, by Definition 3.1, $s\gamma\!\downarrow_{S_E}$ $[\theta] \overset{\emptyset\otimes_{s\gamma} P'}{\underset{i}{\otimes\!\!\Longrightarrow}} s\gamma\!\downarrow_{S_E}$ $[\sigma]$ and $t\gamma\!\downarrow_{S_E}$ $[\theta] \overset{\emptyset\otimes_{t\gamma} Q'}{\underset{j}{\otimes\!\!\Longrightarrow}} t\gamma\!\downarrow_{S_E}$ $[\tau]$.

By Critical pair assumption 1, there is a strongly decreasing diagram for the nested critical peak $s \overset{O}{\underset{i}{\Longleftarrow\!\!\otimes}} r \overset{O'}{\underset{j}{\otimes\!\!\Longrightarrow}} t$, hence $s \rightrightarrows\!<\!i; j\!>\!\rightrightarrows m \leftleftarrows\!<\!i; j\!>\!\leftleftarrows t$. Note that m is normal by definition of an SDR, but, again, $m\gamma$ need not be. By stability of SDRs $s\gamma\!\downarrow_{S_E} \rightrightarrows\!<\!i; j\!>\!\rightrightarrows m\gamma\!\downarrow_{S_E} \leftleftarrows\!<\!i; j\!>\!\leftleftarrows t\gamma\!\downarrow$ is a strongly decreasing diagram as well.

Because θ is preserving, all our rewrite relations are stable, hence $s\gamma\!\downarrow_{S_E}[\theta] \rightrightarrows\!<\!i; j\!>\!\rightrightarrows m\gamma\!\downarrow_{S_E}[\theta] \leftleftarrows\!<\!i; j\!>\!\leftleftarrows t\gamma\!\downarrow_{S_E}[\theta]$. By Corollary 3.7(LAP), $s\gamma\!\downarrow_{S_E}[\sigma] \rightrightarrows\!<\!i; j\!>\!\rightrightarrows m\gamma\!\downarrow_{S_E}[\sigma]$ $\overset{P_2}{\underset{i}{\Longleftarrow\!\!\otimes}} m\gamma\!\downarrow_{S_E}[\theta]$ and $t\gamma\!\downarrow_{S_E}[\tau] \rightrightarrows\!<\!j; i\!>\!\rightrightarrows m\gamma\!\downarrow_{S_E}[\tau] \overset{Q_2}{\underset{j}{\Longleftarrow\!\!\otimes}}$ $m\gamma\!\downarrow_{S_E}[\theta]$. This yields a new local peak $m\gamma\!\downarrow_{S_E}[\sigma] \overset{P_2}{\underset{i}{\Longleftarrow\!\!\otimes}}$ $m\gamma\!\downarrow_{S_E}[\theta] \overset{Q_2}{\underset{j}{\otimes\!\!\Longrightarrow}} m\gamma\!\downarrow_{S_E}[\tau]$, which is strictly smaller than the one we started from.

By induction hypothesis, the local peaks $x\sigma \overset{(P_2)_x}{\underset{i}{\Longleftarrow\!\!\otimes}} x\theta \overset{(Q_2)_x}{\underset{j}{\otimes\!\!\Longrightarrow}} x\tau$ have SDDs $x\sigma \rightrightarrows\!<\!i; j\!>\!\rightrightarrows x\varphi \leftleftarrows\!<\!i; j\!>\!\leftleftarrows x\tau$ for some substitution φ. We have now got a succession of two SDRs from v: $v =_E s\gamma\!\downarrow_{S_E}[\sigma] \rightrightarrows\!<\!i; j\!>\!\rightrightarrows m\gamma\!\downarrow_{S_E}[\sigma] \rightrightarrows\!<\!i; j\!>\!\rightrightarrows m\gamma\!\downarrow_{S_E}[\varphi]$. By Lemma 2.9, the initial $=_E$-step can be eliminated to give an SDR $v \rightrightarrows\!<\!i; j\!>\!\rightrightarrows m\gamma\!\downarrow_{S_E}[\sigma]$. Lemma 5.8 then provides us with an SDR $v \rightrightarrows\!<\!i; j\!>\!\rightrightarrows m\gamma\!\downarrow_{S_E}[\varphi]$. Likewise, there is an SDR $w \rightrightarrows\!<\!j; i\!>\!\rightrightarrows m\gamma\!\downarrow_{S_E}[\varphi]$. Together, these SDRs constitute a strongly decreasing diagram for the starting peak.

We now consider the general case where $O = (P \cup Q)_{min} = \{o_k\}_k$, $P_k = \{p \in P : p \geq_{\mathcal{P}} o_k\}$, $Q_k = \{q \in Q : q \geq_{\mathcal{P}} o_k\}$, and $O_k = P_k \cup Q_k$. We can then apply the previous result to each O_k and then combine the resulting strongly decreasing diagrams for peaks $v \overset{\langle 3,i,P_k\rangle}{\underset{R_{ho}}{\Longleftarrow\!\!\otimes}} u \overset{\langle 3,j,Q_k\rangle}{\underset{R_{ho}}{\otimes\!\!\Longrightarrow}} w$ by using (DP) and Lemma 5.8 to get a strongly decreasing diagram. Here, the facing i-steps (resp. j-steps) will be combined into a single i-step (resp. j-step) since they occur at parallel positions.

Corollary 5.14. *Homogeneous higher-order local peaks have decreasing diagrams under Critical peak assumption 1.*

5.7 DDs for first-order/higher-order local peaks and cliffs

Because pattern matching the lefthand side of a higher-order rule preserves its top function symbol –which belongs to Σ_{ho} by assumption 5–, it cannot overlap the lefthand side of a first-order rule which is entirely built from the first-order signature.

Simplification rules can of course overlap higher-order rules at strict subterms, but they need not be taken into account since higher-order rewriting uses pattern matching modulo the confined theory. Critical pairs are therefore obtained, as for rewriting modulo, by unifying modulo SE the left-hand side l of a rule $l \to r \in R_{fo}$ with a strict subterm $L|_p$ of a rule $L \to R \in R_{ho}$. A key property originating from the definition of higher-order rewriting is that no equation from E can take place on the path from its root to the position p when firing the rule $L \to R$, a crucial condition for these critical pairs modulo to make sense.

Critical peak assumption 2: Critical peaks of first-order rules onto higher-order rules have strongly decreasing diagrams.

Let $v \overset{\langle 1,m,p\rangle}{\underset{j:l\to r\in R_{fo}}{\longleftarrow\!\!\!\!\cdot}} u \overset{\langle 3,i,Q\rangle}{\underset{i:L\to R\in R_{ho}}{\otimes\!\!\Longrightarrow}} w$. The construction of a decreasing diagram is by induction on the size of u modified as before. Let $P = (Q \cup \{p\})_{min}$. We consider first the case where P is a singleton. There are three cases:

- $P = \{p\}$. No overlap of L onto l is possible, as already noticed.

  Let $s = u[\overline{z}]_{Q_{min}}$ for fresh $\overline{z}$ and $s[\sigma] \overset{\text{def}}{=} u$. Since non-confined variables in l are linear, s rewrites at p with rule j.

  By Lemma 3.3, $Q = \emptyset \otimes_s Q'$. Therefore, by Lemma 3.8, $v = t[\sigma] \overset{p}{\underset{j}{\longleftarrow}} s[\sigma] \overset{\emptyset\otimes_s Q'}{\underset{i}{\otimes\!\!\Longrightarrow}} s[\tau] = w$. Hence $t[\sigma] \overset{Q''}{\underset{i}{\otimes\!\!\Longrightarrow}} t[\tau] \overset{p}{\underset{j}{\longleftarrow}} s[\tau]$ for some Q'' by Corollary 3.7.

- $P = \{q \in Q\}$, and $(Q \setminus \{q\}) \cup \{p\} \geq_{\mathcal{P}} q \cdot F_L^{fun}$, implying that there are no overlap of l onto the i-redex at position q. This case is shown at Figure **??**.

  Let $r \overset{\text{def}}{=} u[\overline{z}]_{((Q\setminus\{q\})\cup\{p\})_{min}}$ and $r[\gamma] \overset{\text{def}}{=} u$, hence r, γ are normal. For each z, $z\gamma$ is headed by a symbol in $\Sigma_{fo} \cup \Sigma_{ho}$, hence γ is preserving. By Lemma 3.3, $Q = \{q\} \otimes_r Q'$. By Lemma 3.8, $r[\gamma] \overset{q}{\underset{i}{\longrightarrow}} s[\gamma] \overset{Q'}{\underset{i}{\otimes\!\!\Longrightarrow}} s[\tau] = w$. Now, $p = q' \cdot p'$ for some $q' \in ((Q \setminus \{q\}) \cup \{p\})_{min}$ (possibly, $p' = \Lambda$). Let $z = r|_{q'}$. Then, $z\sigma \overset{p'}{\underset{j}{\longleftarrow}} z\gamma \overset{Q'_z}{\underset{i}{\otimes\!\!\Longrightarrow}} z\tau$. By induction hypothesis, we have $z\sigma \rightrightarrows\!<\!1, m; 3, i\!>\!\rightrightarrows z\theta \leftleftarrows\!<\!1, m; 3, i\!>\!\leftleftarrows z\tau$, a decreasing diagram for this local peak.

  Let now $\sigma(z) \overset{\text{def}}{=} z\sigma$, $\theta(z) \overset{\text{def}}{=} z\theta$, $\sigma(x) \overset{\text{def}}{=} \gamma(x)$ and $\theta(x) \overset{\text{def}}{=} x\tau$ for $x \neq z$. Rewriting $z\gamma$ to $z\sigma$ with the first-order rule j may of course create $(S_{cd})_{E_{cd}}$-redexes, hence we cannot apply (LAP) here to rewrite from $v = r[\gamma]$ to $s[\sigma]$. Instead, we need to first normalize, which will then enable the higher-order rewriting step at position q in v: $v \underset{(S_{cd})_{E_{cd}}}{\overset{*}{\longrightarrow}} r[\sigma\!\downarrow_{S_E}] \overset{q}{\underset{i}{\longrightarrow}} s[\sigma]$, an SDR. Applying now the permutation lemma 5.8 to the derivation from $r[\sigma]$ to $s[\theta]$ yields a decreasing diagram for the starting peak.

- $P = \{q \neq p\}$ and $p <_{\mathcal{P}} q \cdot F_L$. This case is shown at Figure 5.

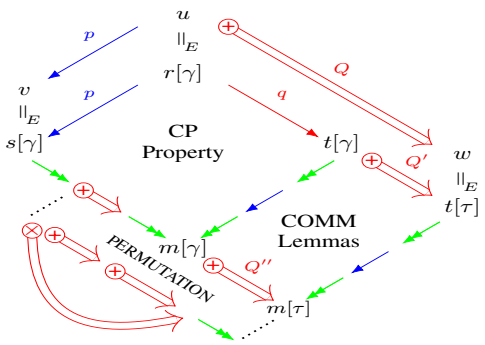

Figure 5. DD for first-order/higher-order overlapping peaks

Let $r \stackrel{\text{def}}{=} u[\overline{z}]_{(Q\setminus\{q\})_{min}}$. By the same token again, $r \xrightarrow[j]{p} s$, hence $r[\gamma] \xrightarrow[j]{p} s[\gamma]$ by stability under substitution of non-confined variables. By Lemma 3.3, $Q = \{q\} \otimes_r Q'$. By Lemma 3.8, $u = r[\gamma] \xrightarrow[j]{q} t[\gamma] \overset{Q'}{\otimes\!\!\Longrightarrow} t[\tau] = w$.

Since symbols at q, p are different by signature assumption, then $p \neq q$. Then, $p \in q\langle \mathcal{FP}os(L) \rangle \setminus (\{\Lambda\} \cup (\geq_{\mathcal{P}} (F_L^{fun} \cup F_L^{cd})))$. Therefore, by definition of higher-order rewriting, $L|_{p'}\theta = u|_p =_E l\theta$ for some θ, p' with $p = q \cdot p'$, assuming L and l share no variable, and therefore $\theta =_E \sigma\delta$, where σ is a most general E-unifier of the equation $L|_{p'} = l$, and $R\sigma \xleftarrow[i]{\Lambda} L\sigma \xrightarrow[j]{p'} L[r\sigma]$ is a critical peak of j onto i. By assumption, this peak has a critical diagram. The proof then follows the case of nested critical peak.

The case where $(P \cup \{p\})_{min}$ is not a singleton set follows the same schema as for HO/HO local peaks, using now (DP).

Corollary 5.15. *Heterogeneous first-order/higher-order local peaks have decreasing diagrams under Critical peak assumption 2.*

5.8 DDs for functional/higher-order local peaks

We omit here the treatment of plain β^0-steps which are a particular case of that of parallel β-step.

Let $v \overset{\langle 2,n,P\rangle}{\underset{\beta}{\Longleftarrow}} u \overset{\langle 3,i,Q\rangle}{\underset{R_{ho}}{\otimes\!\!\Longrightarrow}} w$. By induction on $|P|$, we prove the existence of a decreasing diagram of the form $v \overset{Q'}{\underset{i}{\otimes\!\!\Longrightarrow}} t \underset{\beta}{\Longleftarrow} w$. Since β-rewrites have a smaller category, this diagram is decreasing. By definition of parallel rewriting, $u \xrightarrow[\beta]{p} v' \overset{P\setminus\{p\}}{\underset{\beta}{\Longrightarrow}} v$. Assume the result is true for a single beta-reduction considered as a parallel one. Then $v' \overset{Q'}{\underset{i}{\otimes\!\!\Longrightarrow}} s \underset{\beta}{\Longleftarrow} w$. We then conclude by induction hypothesis.

We now sketch the case of a single beta-rewrite from u at some position p, and assume that $p \geq_{\mathcal{P}} q$, where q is the position of a higher-order rewrite $L \to R$. Since lefthand sides of higher-order rules are beta-normal, p must be below the fringe of L. Then, by Lemma 2.17, the position p is preserved by the higher-order rewrite at q. If instead $q >_{\mathcal{P}} p$, we know that q is preserved by the rewrite at p. It follows that the situation is very much like the first-order/higher-order heterogeneous peaks, with the exception that a beta-rewrite at p may stack redexes occurring below p. This is the reason for introducing orthogonal higher-order rewrites.

Corollary 5.16. *Heterogeneous functional/higher-order local peaks have decreasing diagrams.*

5.9 The confluence theorem

Theorem 5.1. *$\lambda\Pi$Mod equipped with a set of rules and equations (R_{fo}, S_{cd}, E_{cd}, R_{ho}) satisfying all our assumptions is Church-Rosser modulo $\alpha \cup E_{cd}$ on untyped terms.*

6. Encoding Coq's universes in $\lambda\Pi$Mod

Encodings follow a well-understood schema [12]: statements and proofs in a given system are encoded in Dedukti so as to preserve the provability relationship. Further, encodings must be shallow so as to get proof terms. This is especially easy in Dedukti thanks to the user-defined syntax and rewrite rules.

The signature. The encoding uses function symbols to represent sorts, types and terms of CIC. Knowledge of $\text{CCU}_{\subseteq}^{\infty}$ is assumed to understand the rules. Appropriate expositions are [2, 3, 26].

The specification has 3 type constructors, Sort, U and T. The name Sort comes from PTSs. Other declared symbols allow us to build objects and the dependent types they inhabit.

Sort is the type for universes in our encoding, starting with the impredicative universe Prop and continuing with the predicative ones. For convenience, we call them $0, 1, \ldots$, so as to be represented by a copy of the natural numbers generated by three constructors, 0, 1 and $+$. This perhaps uncommon presentation of the natural numbers is instrumental in obtaining a finite Church-Rosser system. The symbols U and T represent, respectively, the type of *codes* and the *decoding function* of universes. Finally, $\mathrm{u}(i)$, $\uparrow(i,a)$, $\Uparrow(i,a)$ and $\pi(i,j,a,b)$ are codes for respectively, the type $\mathrm{U}(i+1)$, the type lifted one level from some type $\mathrm{U}(i)$, the type lifted n level from the type $\mathrm{U}(0)$, and for Π-types.

Variables i, j are of type Sort. Variables a and b are used as the third and fourth argument of π respectively.

$$
\begin{array}{lcl}
\mathrm{Sort} & : & * \\
0, 1 & : & \mathrm{Sort} \\
+^2 & : & \Pi i : \mathrm{Sort} \,.\, \Pi j : \mathrm{Sort} \,.\, \mathrm{Sort} \\
\max^2 & : & \Pi i : \mathrm{Sort} \,.\, \Pi j : \mathrm{Sort} \,.\, \mathrm{Sort} \\
\mathrm{rule}^2 & : & \Pi i : \mathrm{Sort} \,.\, \Pi j : \mathrm{Sort} \,.\, \mathrm{Sort} \\
\mathrm{U}^1 & : & \Pi i : \mathrm{Sort} \,.\, * \\
\mathrm{T}^2 & : & \Pi i : \mathrm{Sort} \,.\, \Pi a : \mathrm{U}(i) \,.\, * \\
\mathrm{u}^1 & : & \Pi i : \mathrm{Sort} \,.\, \mathrm{U}(i+1) \\
\uparrow^2 & : & \Pi i : \mathrm{Sort} \,.\, \Pi a : \mathrm{U}(i) \,.\, \mathrm{U}(i+1) \\
\Uparrow^2 & : & \Pi i : \mathrm{Sort} \,.\, \Pi a : \mathrm{U}(0) \,.\, \mathrm{U}(i) \\
\pi^4 & : & \Pi i : \mathrm{Sort} \,.\, \Pi j : \mathrm{Sort} \,.\, \Pi a : \mathrm{U}(i) \,. \\
 & & \quad \Pi b : (\Pi x : \mathrm{T}(i,a) \,.\, \mathrm{U}(j)) \,.\, \mathrm{U}(\mathrm{rule}(i,j))
\end{array}
$$

The rewrite system. The constructor $+$ is associatif, commutatif, and has 0 as identity element. Since it is a constructor, that is, it never occurs as the head of a lefthand side of rules, no rule extension is needed [15]. We take the liberty to use an infix notation for $+$, the abbreviation 2 for $1+1$, and a varyadic number of arguments to ease the readability of sums.

$$
\begin{array}{lllll}
1: & \max(i, i+j) & \xrightarrow[m_1]{} & i+j \\
2: & \max(i+j, j) & \xrightarrow[m_2]{} & i+j \\
3: & \mathrm{rule}(i, 0) & \xrightarrow[m_3]{} & 0 \\
4: & \mathrm{rule}(i, j+1) & \xrightarrow[r_1]{} & \max(i, j+1) \\
5: & \Uparrow(0, a) & \xrightarrow[l_1]{} & a \\
6: & \Uparrow(i+1, a) & \xrightarrow[l_2]{} & \uparrow(i, \Uparrow(i, a))
\end{array}
$$

The rules for $\max$, rule and $\Uparrow$ are self explanatory. Note that the rewrite rules for rule encodes the impredicativity of Prop, encoded by 0 of type Sort. The name rule comes from PTSs again.

$$
\begin{array}{lll}
7: & \pi(i+1,i+j+1,\uparrow(i,a),b) & \underset{p_1}{\to} \quad \pi(i,i+j+1,a,b) \\
8: & \pi(i+j+2,j+1,\uparrow(i+j+1,a),b) & \\
 & \qquad \underset{p_2}{\to} \quad \uparrow(i+j+1,\pi(i+j+1,j+1,a,b)) & \\
9: & \pi(i+j+2,j+2,a,\lambda x:T(i+j+2,a).\uparrow(j+1,(b\ x))) & \\
 & \qquad \underset{p_3}{\to} \quad \pi(i+j+2,j+1,a,b) & \\
10: & \pi(i,i+j+1,a,\lambda x:T(i+j+1,a).\uparrow(i+j,(b\,x))) & \\
 & \qquad \underset{p_4}{\to} \quad \uparrow(i+j,\pi(i,i+j,a,b)) & \\
11: & \pi(i+1,1,a,\lambda x:T(1,a).\uparrow(0,(b\ x))) & \\
 & \qquad \underset{p_5}{\to} \quad \Uparrow(i+1,\pi(i+1,0,a,b)) & \\
12: & \pi(0,1,a,\lambda x:T(1,a).\uparrow(0,(b\ x))) & \\
 & \qquad \underset{p_6}{\to} \quad \uparrow(0,\pi(0,0,a,b)) & \\
13: & \pi(i+1,0,\uparrow(i,a),b) & \underset{p_7}{\to} \quad \pi(i,0,a,b)
\end{array}
$$

The rules for π are delicate and chosen so as to ensure that types have a unique encoding, a property that is crucial for showing the preservation of typing [2, 3]. Their design obtained by comparing (via $+$) the first two arguments of π ensures that they have few critical pairs (eliminating them all would require a richer language involving a comparison operator and conditional rewrite rules – whose confluence checking is surely less understood. On the other hand, the complex type of b which depends on both first arguments of π impacts their formulation as explained now.

In rules 7, 8 and 13, the first argument of π decreases by 1 in the recursive call while the second argument is unchanged. Typing these rules requires then using conversions with rule 15. For an exemple, consider rule 7. The type of π tells us that in the lefthand side, $b : \Pi x : T(i+1,\uparrow(i,a)).U(i+j+1)$. In the righthand side, we get $b : \Pi x : T(i,a).U(i+j+1)$. These two types are different, but convertible thanks to rule 15.

In rules 9, 10, 11 and 12, the second argument of π decreases by 1 in the recursive call while the first remains unchanged. Typing these rules requires having an explicit abstraction in the lefthand side, so that b can have the appropriate type in the righthand side.

We are left with the rules for T, which are standard decoding rules whose lefthand and righthand sides have type $*$:

$$
\begin{array}{lll}
14: & \mathrm{T}(i+1,\mathrm{u}(i)) & \underset{t_1}{\to} \quad \mathrm{U}(i) \\
15: & \mathrm{T}(i+1,\uparrow(i,a)) & \underset{t_2}{\to} \quad \mathrm{T}(i,a) \\
16: & \mathrm{T}(i,\Uparrow(i,a)) & \underset{t_3}{\to} \quad \mathrm{T}(0,a) \\
17: & \mathrm{T}(0,\pi(i,0,a,b)) & \underset{t_4}{\to} \quad \Pi x:T(i,a).T(0,(b\ x)) \\
18: & \mathrm{T}(i+j,\pi(i,i+j,a,b)) & \underset{t_5}{\to} \quad \Pi x:T(i,a).T(i+j,(b\ x)) \\
19: & \mathrm{T}(i+j+1,\pi(i+j+1,j+1,a,b)) & \underset{t_6}{\to} \quad \Pi x:T(i+j+1,a).T(j+1,(b\ x))
\end{array}
$$

These rewrite rules have been obtained with the crucial help of MAUDE. They differ slightly from those presented in [3], where an infinite number of symbols is used.

The symbols $\mathrm{Sort}, 0, 1, +, \max, \mathrm{rule}, \mathrm{U}, \mathrm{T}, \mathrm{u}, \pi$ and the rules $1-4, 14, 17-19$ define an embedding of the system CCU$^\infty$ (the calculs of constructions with universes but without cumulativity). Since terms without universe variables in the embedding presented here are clearly in exact correspondence with those in the embedding presented there, Theorem 5.2.11 and 6.2.27 of [3] prove that this embedding is faithfull, that is, there are functions $[M]_A$ and $[\![A]\!]$ that faithfully and conservatively translate the terms of CCU$^\infty_\subseteq$ into the terms of $\lambda\Pi$IMod with this signature.

The symbols $\uparrow, \Uparrow$ and the rules $5{-}13, 15{-}16$ add the cumulativity to the previous system. Therefore, according to Theorem 9.2.16 of [3], the obtained full system is an embedding of CCU$^\infty_\subseteq$ which faithfullness is still a very likely conjecture tackled in a forthcoming paper.

A Church-Rosser encoding. This set of rules (and equations for $+$) has been obtained with the crucial help of the MAUDE system by hiding the higher-order aspects of some of the rules, as described in Appendix where the MAUDE input file is given. Finding the appropriate arithmetic (the first 6 rules) for universes was the most delicate part. The other rules were either given or resulted from the use of MAUDE's confluence checker.

We now show that these rules satisfy our assumptions.

First, we split the signature and rules according to our categories. The rules for Π and T contain functional symbols, they are higher-order. Since the other rules contain no functional symbols, nor Π, T, they can be taken as being first-order. Our choice is then to take $\Sigma_{cd} = \{0, 1, +, \max, \mathrm{rule}\}$, $\Sigma_{fo} = \{\uparrow, \Uparrow\}$, and $\Sigma_{ho} = \{\mathrm{T}, \pi, \mathrm{u}, \mathrm{U}\}$.

Assumptions 3 and 4 are then satisfied as well. For Assumption 2, we must show termination in AC equivalence classes and check that the ACZ-critical pairs are joinable by rewriting. Termination can be achieved easily by Rubio's fully syntactic AC path ordering [25]. This order works much like Dershowitz's recursive path ordering, but has a specific status for AC operators that performs additional monotonicity checks. All symbols can have a multiset status, but rule whose status must be lexicographic while the precedence can be $\mathrm{rule} > \max > + > 1 > 0$. Routine calculations show termination. Checking the CPs of R_{fo} is routine too, there is only a trivial one between the two $\max$ rules.

We are left with the assumptions about the higher-order rules.

Apart from a, b, all variables in the higher-order rules are of type Sort, hence are confined. While b occurs linearly in their lefthand sides, a occurs non-linearly in lefthand sides of rules 9-12, hence violating Assumption 5. Since types need only be convertible in $\lambda\Pi$IMod, we replace these rules by rules 9' to 12' obtained by moving the type expression in a condition, for example:

$$
\begin{array}{ll}
9': & \pi(i+j+2,j+2,a,\lambda x:Y.\uparrow(j+1,(b\ x))) \underset{p_3}{\to} \\
 & \quad \pi(i+j+2,j+1,a,b) \textbf{ if } Y == T(i+j+2,a)
\end{array}
$$

where $==$ is interpreted as convertibility. Since any rewrite operating on $T(i+j+2,a)$ preserves convertibility, confluence of the conditionnal typed rules reduces to the confluence of the untyped rules 9" to 12", where these conditions are dropped, as shown with:

$$
\begin{array}{ll}
9'': & \pi(i+j+2,j+2,a,\lambda x:Y.\uparrow(j+1,(b\ x))) \underset{p_3}{\to} \\
 & \quad \pi(i+j+2,j+1,a,b)
\end{array}
$$

What this informal argument shows is that replacing the types of the bound variables in the rules is faithful with respect to confluence, that is, the confluence of the transformed rules implies the confluence of the typed rules.

We are left with our last assumption that the higher-order critical pairs, as well as the mixed higher-order/first-order pairs, of these modified rules have (strongly) decreasing diagrams. All these critical pairs have been computed with MAUDE, while their decreasing diagrams have been computed by hand. These computations are summerized in Figures 6 and 7. The first column lists the critical pairs as given by MAUDE (see Appendix), the second column lists the real overlaps, the third the joinability diagram for the corresponding critical pair, and the last the constraints generated on the rules' indices in order to obtain a decreasing diagram.

Note that the higher-order rewrites with rules 17, 18 and 19, for which $(b\ \ x)$ occur in the righthand side, cause a beta^0-normalization step in case the variable b is instantiated by an ab-

Equation	Critical top overlap
$p_1 = p_3$	$\pi(i+2, i+2, \uparrow(i+1, a), \lambda x : Y.\uparrow(i+1, (b\ x)))$
$p_2 = p_3$	$\pi(i+j+3, i+2, \uparrow(i+j+1, a),$ $\lambda x : Y.\uparrow(i+1, (b\ x)))$
$p_4 = p_1$	$\pi(i+1, i+j+2, \uparrow(i, a),$ $\lambda x : Y.\uparrow(i+j+1, (b\ x)))$
$p_5 = p_1$	$\pi(1, 1, \uparrow(0, a), \lambda x : Y.\uparrow(0, b))$
$p_5 = p_2$	$\pi(i+j+1, 1, \uparrow(i+1, a), \lambda x : Y.\uparrow(0, b))$
$t_4 = t_5$	$\mathrm{T}(0, \pi(0, 0, a, b))$
$t_5 = t_6$	$\mathrm{T}(j+1, \pi(j+1, j+1, a, b))$

Equation	Critical subterm overlap
$\mathrm{T}(0, l_1) = t_3$	$\mathrm{T}(0, \Uparrow(0, a))$
$\mathrm{T}(i+1, l_2) = t_3$	$\mathrm{T}(i+1, \Uparrow(i+1, a))$
$\mathrm{T}(0, p_7) = t_4$	$\mathrm{T}(0, \pi(i+1, 0, \uparrow(i, a), b)$
$\mathrm{T}(i+j+1, p_1) = t_5$	$\mathrm{T}(i+j+1,$ $\pi(i+1, i+j+1, \uparrow(i, a), b))$
$\mathrm{T}(i+2, p_3) = t_5$	$\mathrm{T}(i+2, \pi(i+2,$ $i+2, a, \lambda x : Y.\uparrow(i+1, (b\ x))))$
$\mathrm{T}(i+j+1, p_4) = t_5$	$\mathrm{T}(i+j+1, \pi(i,$ $i+j+1, a, \lambda x : Y.\uparrow(i+j, (b\ x))))$
$\mathrm{T}(1, p_5) = t_5$	$\mathrm{T}(1, \pi(1, 1, a, \lambda x : Y.\uparrow(0, b)))$
$\mathrm{T}(i+1, p_1) = t_6$	$\mathrm{T}(i+1, \pi(i+1, i+1, \uparrow(i, a), b))$
$\mathrm{T}(i+j+2, p_2) = t_6$	$\mathrm{T}(i+j+2, \pi(i+j+2,$ $j+1, \uparrow(i+j+1, a), b))$
$\mathrm{T}(i+j+2, p_3) = t_6$	$\mathrm{T}(i+j+2, \pi(i+j+2,$ $j+2, a, \lambda x : Y.\uparrow(j+1, (b\ x))))$
$\mathrm{T}(i+1, p_5) = t_6$	$\mathrm{T}(i+1,$ $\pi(i+1, 1, a, \lambda x : Y.\uparrow(0, (b\ x))))$

Figure 6. Critical overlaps

Equation	Joinability
$p_1 = p_3$	$\underset{p_4}{\longrightarrow} = \underset{p_2}{\longleftarrow}$
$p_2 = p_3$	$\underset{p_3}{\longrightarrow} = \underset{p_2}{\longleftarrow}$
$p_4 = p_1$	$\underset{p_1}{\longrightarrow} = \underset{p_4}{\longleftarrow}$
$p_5 = p_1$	$\underset{l_2}{\longrightarrow}\underset{l_1}{\longrightarrow}\underset{p_7}{\longrightarrow} = \underset{p_6}{\longleftarrow}$
$p_5 = p_2$	$\underset{l_2}{\longrightarrow}\underset{p_7}{\longrightarrow} = \underset{p_5}{\longleftarrow}$
$t_4 = t_5$	$=$
$t_5 = t_6$	$=$
$T(0, l_1) = t_3$	$=$
$T(i+1, l_2) = t_3$	$\underset{t_2}{\longrightarrow}\underset{t_3}{\longrightarrow} =$
$T(0, p_7) = t_4$	$\underset{t_4}{\longrightarrow} = \underset{t_2}{\longleftarrow}$
$T(i+j+1, p_1) = t_5$	$\underset{t_5}{\longrightarrow} = \underset{t_2}{\longleftarrow}$
$T(i+2, p_3) = p_5$	$\underset{t_5}{\longrightarrow} =$
$T(i+j+1, p_4) = t_5$	$\underset{t_2}{\longrightarrow}\underset{t_5}{\longrightarrow} = \underset{t_2}{\longleftarrow}$
$T(1, p_5) = t_5$	$\underset{l_2}{\longrightarrow}\underset{l_1}{\longrightarrow}\underset{t_2}{\longrightarrow}\underset{t_4}{\longrightarrow} = \underset{t_2}{\longleftarrow}$
$\mathrm{T}(i+1, p_1) = t_6$	$\underset{t_5}{\longrightarrow} = \underset{t_2}{\longleftarrow}$
$\mathrm{T}(i+j+2, p_2) = t_6$	$\underset{t_2}{\longrightarrow}\underset{t_6}{\longrightarrow} = \underset{t_2}{\longleftarrow}$
$\mathrm{T}(i+j+2, p_3) = t_6$	$\underset{t_6}{\longrightarrow} = \underset{t_2}{\longleftarrow}$
$\mathrm{T}(i+1, p_5) = t_6$	$\underset{t_3}{\longrightarrow}\underset{t_4}{\longrightarrow} = \underset{t_2}{\longleftarrow}$

A partial order making all these local peaks decreasing is the following: $p_1 > p_{i\neq 1} > l_j > t_k$.

Figure 7. Decreasing diagrams for critical pairs

straction. This happens when the corresponding lefthand sides are overlapped with rule 9 or 10, whose lefthand side's fourth argument is indeed the abstraction $\lambda x : Y.(b\ x)$.

We also like to point out that the SDDs obtained for many of these critical paires are not developments, in addition to be originating from both sides of the CP. This is in particular the case of the SDD obtained for the critical pair $T(1, p_5) = t_5$.

It should finally be noted that MAUDE is the only available, maintained system which implements (normal) rewriting modulo associativity, commutativity and idempotency. We tried MAUDE with identity used as a rule, but were not able to obtain a confluent system that way. This sophistication has a pratical impact: the need to implement more general rewriting mechanisms in Dedukti in order to type-check Coq-proofs using universe polymorphism.

Theorem 6.1. *The dependent type theory* $\lambda\Pi$Mod *equipped with the encoding of the cumulative hierarchy of predicative universes is Church-Rosser.*

7. Conclusion

We have described a complex, powerful confluence result for dependent type theories, which, like the one underlying Dedukti, admits user-defined rewrite rules. Our target accumulates difficulties: rewriting modulo an algebraic theory, higher-order rewriting based on higher-order pattern matching, combinations of these, non-linear variables in both the lefthand and righthand side of rules, and critical pairs modulo various theories. Many of these features *alone* are new in the context of β-rewriting untyped terms. The latter two, in particular, are known to raise important difficulties regarding confluence tests.

All these advances were however needed by our example of encoding a cumulative hierachy of universes in a dependent type theory. The solutions were actually dictated by a careful analysis of the example itself, especially the notion of confinement which would hardly have popped up otherwise.

Along this work, we kept adding assumptions that we believe are non-essential, in order to facilitate technicalities, in particular the proof of the confluence theorem. The assumptions that we know or believe are non-essential are the following:

1. There is a single non-terminal in the grammar describing the set of untyped terms. Having several would not make any change.

   Further, we could have $(M\ C)$ and $\lambda y : C.N$ both together as (non-confined) expressions in the grammar. It turns out that this might be needed to add explicit universe polymorphism in Dedukti itself.

2. First-order equations and simplifiers are built from the confined sub-signature and there are no higher-order equations. There could be additional non-confined equations in the first-order signature, this extension should be slightly more complex to carry out since equatons would now operate on non-confined sub-expressions. Similarly, adding equations for higher-order expressions, like associativity and commutativity of some higher-order symbol, would impact both the proofs, for the same reason as before, and the algorithmics: higher-order AC-unification would then be needed [10], which is not as bad as it looks like since we have restricted ourselves to linear patterns in lefthand sides of higher-order rules.

3. Non-confined variables appear linearly on both sides of first-order rules. We conjecture that non-confined vaiables may appear non-linearly in righthand sides, but do not know how to adapt the current proof to this weaker assumption.

4. Righthand sides of higher-order rules cannot be variables nor abstractions. Lifting this restriction would require important technical changes. A weaker definition of higher-order rewriting a term u with rule instance $L\gamma \to R\gamma$ in which the β^0-steps from $L\gamma$ to u are exactly those needed to eliminate the head applications of the pre-redexes $(X\gamma\ \ \overline{x})$ should help.
5. More generally, many assumptions aim at preserving rewrites under substitution instance, that is, eliminating the need of normalizations (with β^0_α or $S_{cd\,E_{cd}}$) each time a rewrite step is instantiated (preserving substitutions, strongly decreasing reductions, etc). We believe it is possible to carry around these normalizations in proofs by using the property that the rewrite system (S, E) is Church-Rosser and terminating. This would then force us to use the general version of decreasng diagrams for which rewrites can go both ways.
6. We have not considered the η-rule in this work. Adding it should not raise many difficulties since algebraic symbols have a *fixed arity* in $\lambda\Pi$IMod.

We believe that future encodings will force us to remove or weaken these assumptions in turn.

On the other hand, we doubt that any of the other assumptions can be removed. The termination assumption for R_{fo}, although natural and required for type checking, came in late. It relates to the existence of non-linear (hence confined) variables in both the lefthand and righthand sides of higher-order rules, making it impossible to find decreasing diagrams for some peaks without it.

Finally, one may question our encoding of Coq's universes in the first place. In Coq's implementation, universes are floating. We could mimick floating universes by using rewriting under constraints. Unfortunately, confluence under constraints is hard, although restricting the constraints to confined types would surely help. Similarly, we did not use conditional rules, whose confluence is even more delicate, although this would have simplified our definition of π. Results exist in the first-order case [20], but are very limited in the higher-order one [8].

References

[1] Claus Appel, Vincent van Oostrom, and Jakob Grue Simonsen. Higher-order (non-)modularity. In Christopher Lynch, editor, *Proceedings of the 21st International Conference on Rewriting Techniques and Applications, RTA 2010, July 11-13, 2010, Edinburgh, Scottland, UK*, volume 6 of *LIPIcs*, pages 17–32. Schloss Dagstuhl - Leibniz-Zentrum fuer Informatik, 2010.

[2] Ali Assaf. A calculus of constructions with explicit subtyping. In Hugo Herbelin, Pierre Letouzey, and Matthieu Sozeau, editors, *20th International Conference on Types for Proofs and Programs, TYPES 2014, May 12-15, 2014, Paris, France*, volume 39 of *LIPIcs*, pages 27–46. Schloss Dagstuhl - Leibniz-Zentrum fuer Informatik, 2014.

[3] Ali Assaf. *A framework for defining computational higher-order logics*. PhD thesis, École polytechnique, Paris, 2015.

[4] Ali Assaf and Guillaume Burel. Translating HOL to Dedukti. In Cezary Kaliszyk and Andrei Paskevich, editors, *Proceedings Fourth Workshop on Proof eXchange for Theorem Proving, PxTP 2015, Berlin, Germany, August 2-3, 2015.*, volume 186 of *EPTCS*, pages 74–88, 2015.

[5] Dowek at all. The Dedukti system. Available from http://dedukti.gforge.inria.fr/.

[6] Hendrik Pieter Barendregt. *The lambda calculus : its syntax and semantics*. Studies in logic and the foundations of mathematics. North-Holland, Amsterdam, New-York, Oxford, 1981.

[7] F. Blanqui, J.-P. Jouannaud, and M. Okada. Inductive-data-type systems. *Theoretical Computer Science*, 272:41–68, 2002.

[8] Frédéric Blanqui. Termination of rewrite relations on λ-terms based on Girard's notion of reducibility. *Theor. Comput. Sci.*, 611:50–86, 2016.

[9] Mathieu Boespflug and Guillaume Burel. CoqInE: Translating the calculus of inductive constructions into the lambda-Pi-calculus modulo. In *Proof Exchange for Theorem Proving—Second International Workshop, PxTP*, page 44, 2012.

[10] A. Boudet and E. Contejean. AC-unification of higher-order patterns. In *Int. Conf. on Constraint Programming, 1997*, pages 267–281, 1997.

[11] Manuel Clavel, Francisco Durán, Steven Eker, Patrick Lincoln, Narciso Martí-Oliet, José Meseguer, and Carolyn L. Talcott, editors. *All About Maude - A High-Performance Logical Framework, How to Specify, Program and Verify Systems in Rewriting Logic*, volume 4350 of *Lecture Notes in Computer Science*. Springer, 2007.

[12] Denis Cousineau and Gilles Dowek. Embedding pure type systems in the lambda-pi-calculus modulo. In Simona Ronchi Della Rocca, editor, *Typed Lambda Calculi and Applications, 8th International Conference, TLCA 2007, Paris, France, June 26-28, 2007, Proceedings*, volume 4583 of *Lecture Notes in Computer Science*, pages 102–117. Springer, 2007.

[13] Healfdene Goguen. The metatheory of UTT. In Peter Dybjer, Bengt Nordström, and Jan M. Smith, editors, *Types for Proofs and Programs, International Workshop TYPES'94, Båstad, Sweden, June 6-10, 1994, Selected Papers*, volume 996 of *Lecture Notes in Computer Science*, pages 60–82. Springer, 1994.

[14] G. Huet. Confluent reductions: abstract properties and applications to term rewriting systems. *Journal of the ACM*, 27(4):797–821, October 1980.

[15] J.-P. Jouannaud and H. Kirchner. Completion of a set of rules modulo a set of equations. *SIAM Journal of Computing*, 15(4):1155–1194, 1986.

[16] J.-P. Jouannaud and C. Marché. Termination and completion modulo associativity, commutativity and identity. *Theoretical Computer Science*, 104:29–51, 1992.

[17] Jean-Pierre Jouannaud and Jianqi Li. Church-Rosser properties of normal rewriting. In Patrick Cégielski and Arnaud Durand, editors, *Computer Science Logic (CSL'12) - 21st Annual Conference of the EACSL, CSL 2012, September 3-6, 2012, Fontainebleau, France*, volume 16 of *LIPIcs*, pages 350–365. Schloss Dagstuhl - Leibniz-Zentrum fuer Informatik, 2012.

[18] Jean-Pierre Jouannaud and Jiaxiang Liu. From diagrammatic confluence to modularity. *Theor. Comput. Sci.*, 464:20–34, 2012.

[19] Jan Willem Klop. *Combinatory reduction systems*. PhD thesis, CWI tracts, 1980.

[20] Salvador Lucas and José Meseguer. Normal forms and normal theories in conditional rewriting. *J. Log. Algebr. Meth. Program.*, 85(1):67–97, 2016.

[21] Claude Marché. Normalized rewriting: An alternative to rewriting modulo a set of equations. *J. Symb. Comput.*, 21(3):253–288, 1996.

[22] D. Miller. A logic programming language with lambda-abstraction, function variables, and simple unification. *Journal of Logic and Computation*, 1(4):497–536, 1991.

[23] T. Nipkow. Higher-order critical pairs. In *Proceedings of the 6th annual IEEE Symposium on Logic in Computer Science (LICS '91)*, pages 342–349, Amsterdam, The Netherlands, July 1991.

[24] Ulf Norell. Dependently typed programming in agda. In Pieter W. M. Koopman, Rinus Plasmeijer, and S. Doaitse Swierstra, editors, *Advanced Functional Programming, 6th International School, AFP 2008, Heijen, The Netherlands, May 2008, Revised Lectures*, volume 5832 of *Lecture Notes in Computer Science*, pages 230–266. Springer, 2008.

[25] Albert Rubio. A fully syntactic AC-RPO. *Inf. Comput.*, 178(2):515–533, 2002.

[26] Matthieu Sozeau and Nicolas Tabareau. Universe polymorphism in Coq. In Gerwin Klein and Ruben Gamboa, editors, *Interactive Theorem Proving - 5th International Conference, ITP 2014, Held as Part*

of the Vienna Summer of Logic, VSL 2014, Vienna, Austria, July 14-17, 2014. Proceedings, volume 8558 of *Lecture Notes in Computer Science*, pages 499–514. Springer, 2014.

[27] Terese. Term rewriting systems. In *Cambridge Tracts in Theoretical Computer Science, Marc Bezem, Jan Willem Klop, and Roel de Vrijer editors*. Cambridge University Press, 2003.

[28] Vincent van Oostrom. Confluence by decreasing diagrams. *Theor. Comput. Sci.*, 126(2):259–280, 1994.

[29] Vincent van Oostrom. Developing developments. *Theor. Comput. Sci.*, 175(1):159–181, 1997.

[30] Vincent van Oostrom. Confluence by decreasing diagrams converted. In Voronkov A., editor, *RTA*, volume 5117 of *Lecture Notes in Computer Science*, pages 306–320. Springer, 2008.

Appendix: Using MAUDE

Checking the local confluence of a rewrite system modulo associativity, commutativity, and identity is a long and delicate task that is prone to errors when done manually. We verified the local confluence of our rewrite system with the help of the MAUDE Formal Environment. Since MAUDE only supports first-order rewrite systems, we slightly adapted our rewrite system to hide the higher-order nature of some symbols. This transformation does not impact the computation of critical pairs in the sense that the higher-order critical pairs can then be easily computed from the first order ones computed by MAUDE (there are many more). Here is the resulting MAUDE specification:

```
fmod LPM is
  sort Nat .
  sort Term .
  sort Type .

  op 0 : -> Nat [ctor] .
  op 1 : -> Nat [ctor] .
  op _+_ : Nat Nat -> Nat [comm assoc id: 0 ctor] .
  op max : Nat Nat -> Nat .
  op rule : Nat Nat -> Nat .

  op u : Nat -> Term .
  op lift : Nat Term -> Term .
  op liftn : Nat Term -> Term .
  op pi : Nat Nat Term Term -> Term .

  op U : Nat -> Type .
  op T : Nat Term -> Type .
  op Pi : Type Type -> Type .

  vars i j k l m x : Nat .
  vars a b : Term .

  eq max(i, i + j) = i + j .
  eq max(i + j, j) = i + j .
  eq rule(i, 0) = 0 .
  eq rule(i, j + 1) = max(i, j + 1) .

  eq liftn(0, a) = a .
  eq liftn(k + 1, a) = lift(k, liftn(k, a)) .

  eq pi(i + 1, i + j + 1, lift(i, a), b) =
     pi(i, i + j + 1, a, b) .
  eq pi(i + j + 1 + 1, j + 1, lift(i + j + 1, a), b) =
     lift(i + j + 1, pi(i + j + 1, j + 1, a, b)) .
  eq pi(i + j + 1 + 1, j + 1 + 1, a, lift(j + 1, b)) =
     pi(i + j + 1 + 1, j + 1, a, b) .
  eq pi(i, i + j + 1, a, lift(i + j, b)) =
  = lift(i + j, pi(i, i + j, a, b)) .
  eq pi(i + 1, 1, a, lift(0, b)) =
     liftn(i + 1, pi(i + 1, 0, a, b)) .
  eq pi(0, 1, a, lift(0, b)) = lift(0, pi(0, 0, a, b)) .
  eq pi(i + 1, 0, lift(i, a), b) = pi(i, 0, a, b) .

  eq T(i + 1, u(i)) = U(i) .
  eq T(i + 1, lift(i, a)) = T(i, a) .
  eq T(i, liftn(i, a)) = T(0, a) .
  eq T(0, pi(i, 0, a, b)) = Pi(T(i, a), T(0, b)) .
  eq T(i + j, pi(i, i + j, a, b)) =
     = Pi(T(i, a), T(i + j, b)) .
  eq T(i + j + 1, pi(i + j + 1, j + 1, a, b)) =
     Pi(T(i + j + 1, a), T(j + 1, b)) .
endfm
```

This system can then be checked using the Checker tool of MAUDE: `(select tool CRC .)` and `(ccr LPM .)`.

The versions of the tools we used in our stack are: MAUDE 2.6 (Oct 13 2011) ; Full MAUDE 2.6.1e (July 13th 2012) ; MAUDE Formal Environment 1.0 ; Church-Rosser Checker 3m (July 7th 2012).